

SEPTEMBER 1984 ÖS 50,–/Sfr 6,– DM 6,–

Eine Markt & Technik Publikation



9|84 DAS MAGAZIN FÜR COMPUTER-FANS

C 64 – ein Musikwunder

★ Test: Die besten Musikprogramme
★ Ein Synthesizer zum Abtippen
★ Grundlagen der Tonerzeugung

Maschinensprache leicht gemacht
Assembler-Kurs

Listing des Monats
Spielegenerator
Ein Baukasten der 1000 Möglichkeiten

Es geht doch: 64 KByte RAM für VC 20

Zum Abtippen:
Mailbox für C 64
Jedem sein elektronischer Briefkasten

Kurs: Der gläserne VC 20 ★ Software-Test: Calc Result und Multiplan, Quickcopy ★ Lösungen für Hobbit und Black-pool ★ Super Sprite-Editor, Screen Change und wieder viele Tips und Tricks

INHALT

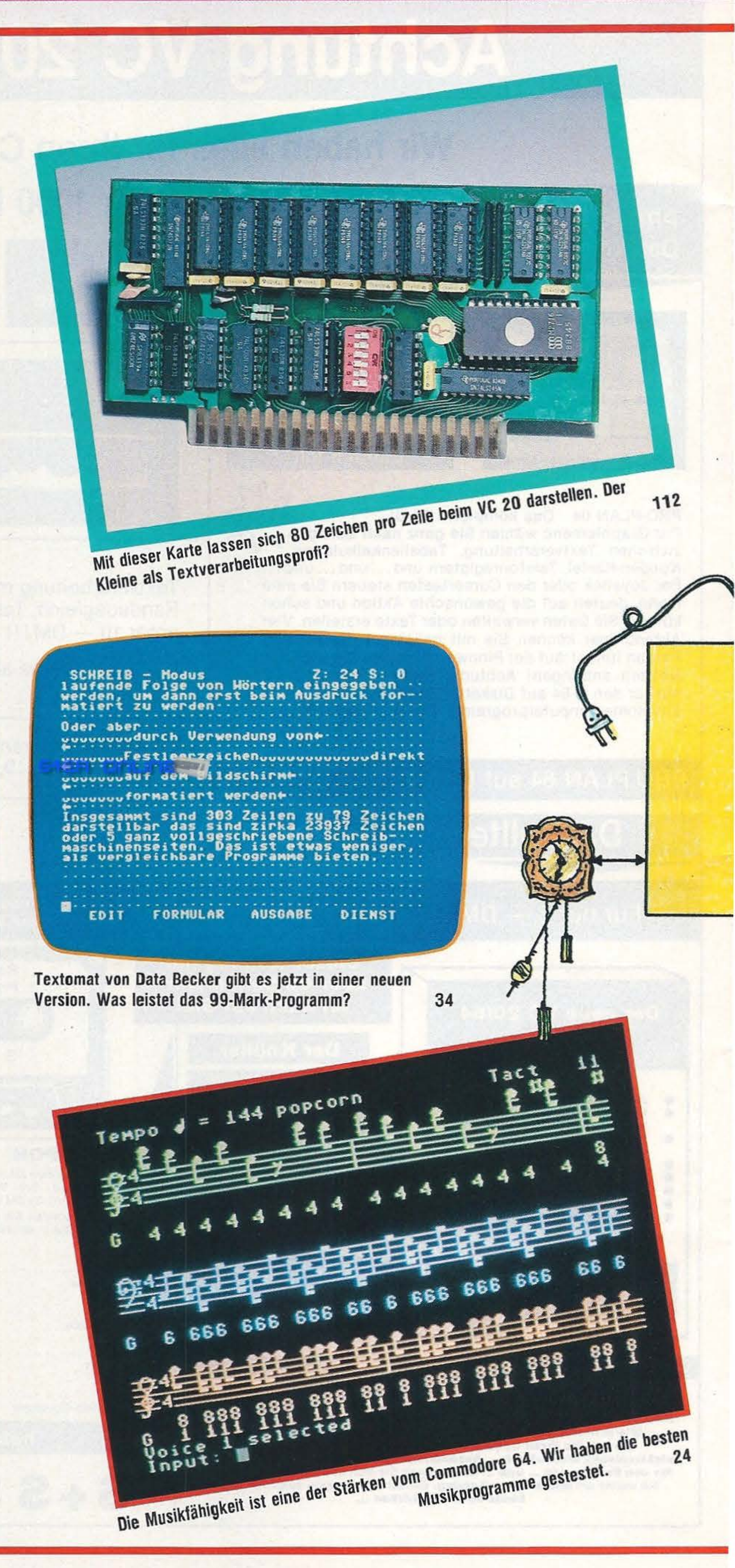
Aktuell

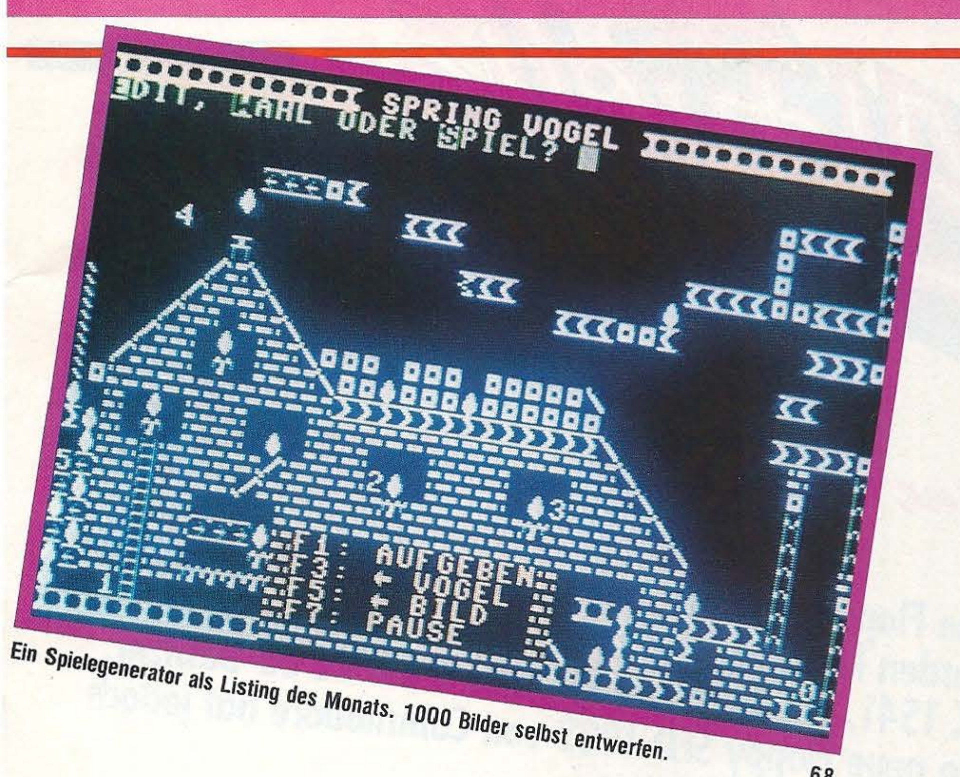
Software

Computer und Musik

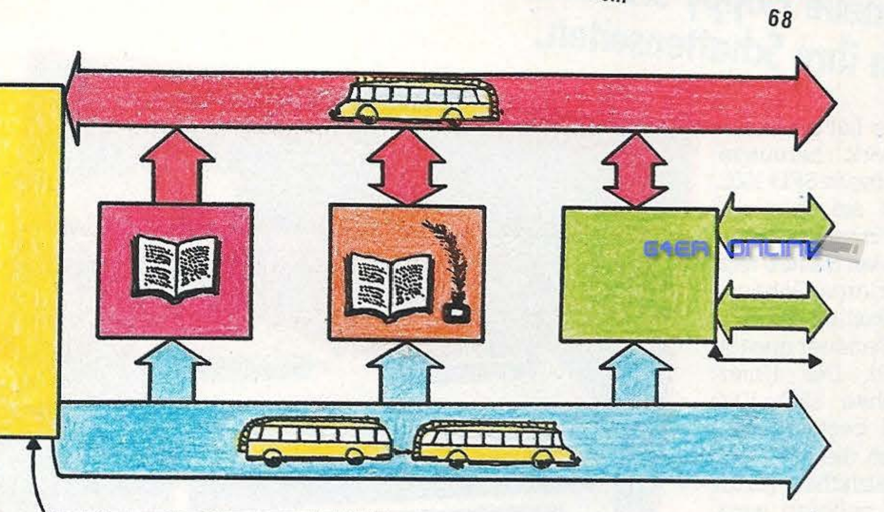
Software-Test

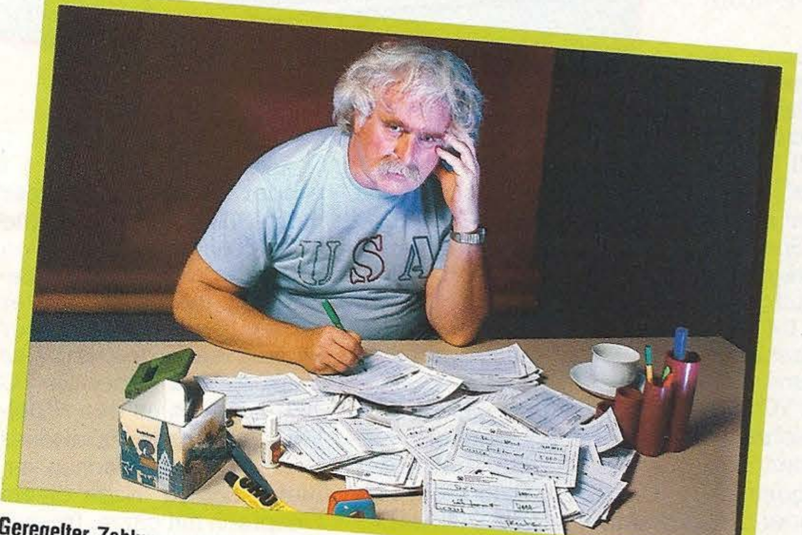
Programme zum Abtippen





Ein Spielegenerator als Listing des Monats. 1000 Bilder selbst entwerfen.
68


Der lang erwartete Assembler-Kurs beginnt in dieser Ausgabe. Steigen Sie ein in die faszinierende Welt der Maschinensprache.
138



Geregelter Zahlungsverkehr: der Kugelschreiber und die Schreibmaschine gehören der Vergangenheit an.
164

Anwendung des Monats

Listing des Monats

Spiele-Test

Hardware-Test

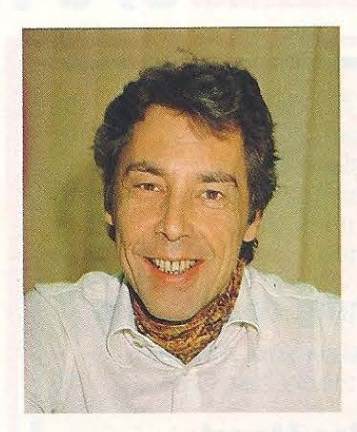
Hardware

Kurse

So machen's andere

Wettbewerbe

Rubriken



Musik lohnt sich

Es ist unverkennbar, daß es in der letzten Zeit einen richtigen Schub gegeben hat — was die Zahl der Musikprogramme für den 64 und vor allem ihre Qualität betrifft. Die interessanteste Software stellen wir in diesem Heft vor. Dabei ist es selbst für Fachleute auf dem Gebiet der elektronischen Musik erstaunlich, was man aus einem Heimcomputer und den vergleichsweise billigen Programmen herausholen kann.

Der Computer kann nicht nur als Musikinstrument dienen — er unterstützt auch das Lernen und Üben, erlaubt die Speicherung und Wiedergabe von Musik, ermöglicht das Zusammenstellen eigener »Sounds« ebenso wie Kompositionsversuche und nimmt keine Experimente übel.

Selbst das Noten-Schreiben kann man dem Computer beziehungsweise dem Drucker überlassen. Vieles, was jetzt jedermann selbst ausprobieren kann, blieb früher auch für musikalisch Interessierte trockene Lehrbuch-Theorie.

Natürlich macht ein 100-Mark-Programm aus einem 64 keinen Konzert-Synthesizer und aus einem schwerhörigen Anfänger kein musikalisches Wunderkind. Aber ein gutes Musikprogramm gehört heute wahrscheinlich zu den lohnendsten Anschaffungen die ein Heimcomputer-Besitzer machen kann — auch wenn die Beschreibungen meist furchtbar technisch und gar nicht melodisch klingen. Ein gutes Spielprogramm ist oftmals auch nicht billiger — wird aber in der Regel sehr viel schneller langweilig. Mit einem branchentypischen Problem müssen Interessenten allerdings rechnen: Die Programme bieten zum Teil schon so viele Möglichkeiten, daß sie wahrscheinlich nur von den wenigsten Verkäufern auch erschöpfend vorgeführt werden können.

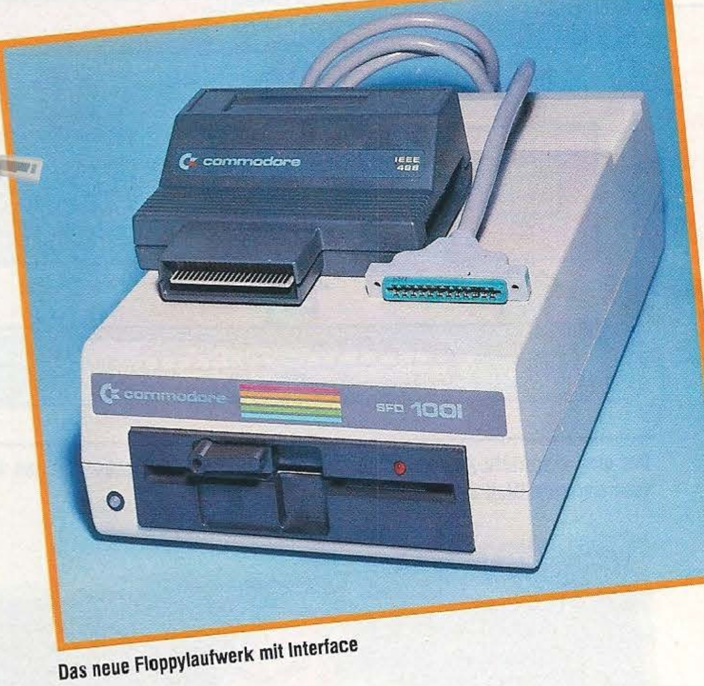
Apropos Titelbild: Fragen Sie bitte nicht, wo man dieses Gerät kaufen kann — das ist natürlich eine Fotomontage.

Michael Pauly, Chefredakteur



Eine neue Floppy für den C 64

Eine Floppy, auf der mehr als ein MegaByte gespeichert werden kann und die zirka viermal schneller ist als die VC 1541, ist ein Traum für viele C 64/VC 20-Besitzer. Die neue Floppy SFD 1002 von Commodore hat jedoch auch ihre Schattenseiten.



Das neue Floppylaufwerk mit Interface

Commodore hat ein neues Floppy-Laufwerk herausgebracht. Ihr Name ist SFD 1002. Eigentlich ist sie ja nichts Neues. Wenn man die große CBM 8250 in zwei Hälften teilt, und in ein anderes Gehäuse hineinsteckt, kommt die SFD 1002 heraus. Genauer gesagt, die SFD 1001. Der Unterschied zwischen SFD 1002 und SFD 1001 besteht lediglich darin, daß die SFD 1001 mit einem zusätzlichem IEEE-488 Interface geliefert wird, das den C 64 mit der 1001 verbindet. Doch kommen wir zu einigen interessanten technischen Daten der SFD 1001/1002.

Nach dem Formatieren einer Diskette meldet die 1001/1002 fantastische 4133 freie Blöcke. Das entspricht umgerechnet 1033,25 KByte freiem Speicherplatz! Daten werden auf 77 Spuren bei 23 bis 29 Sektoren pro Spur gespeichert. Die Übertragungsgeschwindigkeit beträgt 1,2 KByte/sec (VC 1541 : 0,4 KByte/sec) Auch die Anzahl der gleichzeitig geöffneten Dateien ist gegenüber der VC 1541 höher. Es können gleichzeitig fünf sequentielle oder drei relative oder zwei sequentielle und zwei relative oder drei sequentielle und eine relative Datei geöffnet sein.

Die Tatsache, daß sie voll kompatibel zum CBM 8250 Format ist, bedeutet gleichzeitig deren Unverträglichkeit mit der VC 1541. Lassen sich denn Programme von der 1541 auf die 1002 übertragen?

Sie denken vielleicht: kein Problem. Man ändere einfach die Geräteadresse der VC 1541 auf 9, lädt ein Programm mit LOAD»NAME«,9 und speichere es mit SAVE»name«,8 auf der SFD 1001 wieder ab. Das geht aber nicht. Nicht etwa, daß die SFD 1002 die gewohnten Floppybefehle nicht beherrscht, das macht sie. Der Haken liegt woanders: Der serielle Bus ist gesperrt. Das heißt, es läuft weder die VC 1541 noch ein an den seriellen Port angeschlossener Drucker. Das ist schon ein harter Schlag. Nach einem Gespräch mit einem freundlichen Commodore-Fachmann erhielt ich zwei Tage

später ein kleines Programm auf einer Diskette, das dieses Manko etwas lindert. Mit jeweils einem SYS-Befehl kann man auf einen anderen Port umschalten, entweder auf den seriellen oder auf den Expansion-Port. Sie können sich vorstellen, daß das Kopieren einer Diskette auf das SFD 1002 Format zur zermürbenden Prozedur wird.

Dieses Hilfsprogramm soll, so der Spezialist, zukünftig entweder auf der beigelegten Demodiskette gespeichert oder als abzutippendes Listing der Lieferung beigelegt werden.

Spezialkabel nötig

Ein anderes Kapitel ist das Interface. Zusätzlich zum Interface benötigt man ein Spezialkabel vom Interface zur Floppy und ein anderes von der Floppy zu einem weiteren Peripheriegerät (Kosten pro Kabel: zirka 100 Mark; zusätzlich bestellen). Das Interface selbst besitzt oben einen Schacht, in dem Module eingesetzt werden können. An sich eine lobenswerte Einrichtung. Sie hat jedoch einen kleinen Schönheitsfehler: Sobald man ein Modul hineingesteckt hat, funktioniert nichts mehr. Ein echter Konstruktionsfehler. Auch mit Hilfe einer Steckplatzerweiterung, auf die das Interface und ein beliebiges Modul gesteckt werden, bleibt der Bildschirm dunkel und nichts läuft mehr.

Alles in allem kann ich dieses neue Gerät nicht bedingungslos empfehlen. Die Anpassung an den C 64 ist (noch) nicht optimal gelöst. Und für knapp 2000 Mark sollte man Kompatibilität und Portabilität erwarten können. Aber sicherlich werden dementsprechende Lösungen nicht allzulange auf sich warten lassen. Warten wir's ab.

(gk)

COMPUmask

Bei COMPUmask handelt es sich um eine Schablone, die auf den C 64 oder den VC 20 gelegt wird. Auf dieser abtrieb- und reinigungsmittelfesten Schablone sind die am häufigsten vorkommenden Daten, Funktionen, Befehle, Zeichen, POKEs, Tabellen und vieles mehr aufgeführt. Die Oberseite ist zweifarbig, wobei die auf den Bildschirm bezogenen Teile farbig unterlegt sind. So sind über 80 Basic-Befehle, Drucker- und Floppyoperationen, POKEs, Bildschirmausgabezeichen und 3 Demoprogramme zu finden. Diese kleine Übersicht kann nicht alles aufführen, was die Schablone beinhaltet. Für den Preis von 29,80 Mark dürfte die Schablone so für manchen C 64- oder VC 20-Neuling interessant sein. (rg)

Info: IDEE-SOFT, I.Dinkler, Am Schneiderhaus 7, 5760 Arnsberg 1, Tel. 02932/32947



Compumask auf dem C 64

Komfort-Makroassembler für CBM und VC20/C 64

Für die Commodore-Geräte 3032, 4032, 8032, C 64 und VC20 ist das Makroassembler-Paket ASSI/M und ASSI/MC erhältlich. Es besteht aus drei Programmen, dem Fullscreen-Editor FSE, einem Makroassembler ASM für den 6502 oder — im Paket ASSI/MC — die CMOS CPU 65C02 und dem Debugger DEMON beziehungsweise DEMON/C sowie zwei Makrobibliotheken.

Der Assembler übersetzt von Floppy nach Floppy oder von der Floppy direkt in den Speicher, akzeptiert beliebig lange Sourcefiles (mit Verkettung und include) und kann das Listing auf ein beliebiges Gerät ausgeben oder ganz unterdrücken. Im Gegensatz zu anderen Assemblern kann man (wie zum Beispiel bei Pascal) eine Blockstruktur verwenden. Der ASM unterstützt rekursive Makroaufrufe beliebiger Tiefe sowie bis zu 255 Ebenen der Schachtelung bei bedingter Assemblierung. Weitere Eigenschaften: formatfreie Eingabe, lange Symbol-Namen, 23 Klartext-Fehlermeldungen und eine sehr umfangreiche Arithmetik. Der ASM kann Assemblerprogramme verarbeiten, die vom Basic-Compiler BASS der Firma gmb-Soft erzeugt wurden.

Der Editor FSE verwendet keine Zeilennummern, sondern erlaubt Textmanipulationen wie bei Textverarbeitungssystemen mit 2-Richtungs-Scroll, Blättern, frei definierbarem Tabulator und Rändern, Statuszeile, Such- und Austauschbefehlen mit vielen Optionen, Merker, Blockbefehle, arbeitet mit beliebiger Peripherie, auch Kassette, zusammen und bietet noch viele weitere Möglichkeiten.

Der Debugger DEMON bietet Line-Assemblierung, Disassemblierung, Single-Step (auch durch ROM-Bereiche), überwachte Ausführung ohne Anzeige der einzelnen Schritte, fünf Breakpoints, vollständige Kontrolle bei Trace durch frei programmierbare Überwachungsroutine, Arithmetik und Zahlenumwandlungen, Analyse von Programmen auf Verwendung von Zero-Page-Adressen, Verschiebbarkeit von Programmen und ist selbst voll verschiebblich.

Die Makrobibliotheken erlauben die Verwendung von Befehlen für strukturierte Programmierung wie if/then/else, repeat/until, while/endwhile und switch/case/default/endswitch sowie von 16-Bit-Befehlen. Das Paket ist auf Disketten im Format 8050 oder 1541/4040 lieferbar und kostet 220 Mark für den 6502 und 250 Mark für den 65C02.

Info: D. Zabel, Stresemannstr. 50, 1000 Berlin 61, Tel. (030) 8225227



Neue Single-Drive Floppy für Commodore

Auf dem Markt der 5¼-Zoll-Floppylaufwerke für den C 64 und den VC 20 ist ein neuer Konkurrent aufgetaucht. Er heißt YL-55SIL-CM. Auch ein neues Kassettenlaufwerk wird angeboten.



Floppy- und Kassettenlaufwerk von WM-Computer

Diese neue Floppy ist mit einem Teac-Laufwerk ausgestattet, das für seine Zuverlässigkeit bekannt ist. Es ist voll kompatibel zur VC 1541. Das bedeutet, daß mit der VC 1541 bespielte Disketten auch von der neuen Floppy gelesen werden können. Die Geschwindigkeit, die mit diesem Laufwerk erreicht werden könnte, würde das Herz eines jeden VC 1541-Geschädigten höher schlagen lassen. Wenn da nicht noch das Commodore-kompatible DOS wäre. So bringt diese neue Floppy in diesem Punkt keinen Vorteil gegenüber der VC 1541. Laut Hersteller wird aber daran gearbeitet, die Geschwindigkeit, mit der das Laufwerk arbeitet, zu verbessern. Diejenigen, die sich die neue Floppy schon jetzt kaufen möchten, können später das alte DOS gegen einen geringen Aufpreis austauschen lassen. Doch die Übertragungsgeschwindigkeit bei einer Floppy ist nicht alles. Wer die Reparaturanfälligkeit der Commodore-Floppy kennt, wird ein zuverlässigeres Laufwerk schätzen. Der empfohlene Verkaufspreis liegt bei 798 Mark.

99-Mark-Recorder

Der Anbieter der neuen Floppy bringt auch ein Alternativprodukt für die Datasette auf den Markt. Das äußerlich ansprechende Produkt wird zum empfohlenen Richtpreis von 99 Mark erhältlich sein.

Info: Bezugsquelle: WM-Computer, St.-Anton-Str.31, 4150 Krefeld 1, 02151/801300 und Hertie.

Commodore-Messe in Frankfurt

Vom 4. bis 8. September findet in der Halle 1 des Frankfurter Messegeländes die vierte »Internationale Commodore-Fachausstellung (CFA)« statt. Etwa 120 Aussteller präsentieren Hard- und Software rund um den Commodore. Laut Commodore werden beide Schwerpunkte berücksichtigt: kommerziell/professionelle Anwendungen und der Hobby-Bereich. Für »Freaks« gebe es eine spezielle Hard- und Softwarebörse, auf der Commodore-Computer und -Programme verkauft oder getauscht werden können. Übrigens: Die Redaktion des 64'er-Magazins steht während der CFA für Gespräche ihren Lesern zur Verfügung (Stand: Markt & Technik). (kg)

Computer-Bücher von Commodore

Bedienungshandbücher, die beim Kauf eines Computers mitgeliefert werden, sind oftmals nicht mehr als eine relativ lückenhafte Gebrauchsanweisung. Commodore will diese Lücken mit ihrer neuen Sachbuchreihe füllen. Bislang sind fünf Bände erschienen. Band 1: Alles über den Commodore 64 (59 Mark); Band 2: Alles über den VC 20 (24,90 Mark); Band 3: Logo, nur mit der Software erhältlich, Buch in deutsch, Programm in englisch (zusammen 159 Mark); Band 4: Das C 64-Spielebuch (29,80 Mark); Band 5: Das VC 20 Spielebuch (29,80 Mark). Die Reihe soll kontinuierlich fortgesetzt werden. (kg)

Modem als Bausatz

Als fertiges Gerät oder als Bausatz ist dieses neue Modem preiswert zu beziehen. Es hat allerdings keine FTZ-Nummer. Der Bausatz soll zirka 260 Mark kosten, das fertige Gerät dürfte um etwa 100 Mark teurer sein. Zirka 220 Einzelteile müssen Sie zusammenschrauben beziehungsweise löten, bevor Sie das Modem einsetzen können. Dann sollten Sie ein voll einsatzfähiges Gerät haben, das alle Voraussetzungen eines professionellen Modem besitzt.

Bezugsquelle: Software Express, Hugo-Vichoff-Str. 84, 4000 Düsseldorf 30



Das Selbstbau-Modem



Probleme mit Datasette?

Ich habe andauernd »LOAD ERROR« trotz Kassettenwechsel und einwandfrei justiertem Lese- und Aufnahmekopf der Datasette. Kann es am C 64 liegen? Wer kann helfen?

Peter Ritter

Schwierigkeiten mit Farbfernsehen und Interface?

Ich besitze einen C 64 und das Kassetteninterface von K. Jeschke. Früher hatte ich einen Schwarzweiß-Fernseher und das Interface arbeitete einwandfrei. Seit kurzem besitze ich einen kleinen Farbfernseher, doch jetzt tritt ein Problem auf. Das Speichern funktioniert weiterhin einwandfrei, doch beim Laden muß ich jetzt immer erst den Antennenstecker aus dem C 64 ziehen und Programme blind einladen, weil bei angeschlossenem Fernseher das Relais im Interface verrückt spielt und das Einladen nicht funktioniert. Woran kann die Unverträglichkeit zwischen Fernseher und Interface beim Laden liegen?

Andreas Heine

Das Problem liegt wahrscheinlich in einer mangelhaften Abschirmung des HF-Signals in Ihrem Fernseher. Mögliche Abhilfe: Standort des Interfaces verlagern, Anschlußkabel verdrillen. Besser abgeschirmtes Kabel vom Fernseher zum C 64 verwenden (75 Ohm Antennenkabel).

Probleme mit unseren Listings?

Was bedeuten im 64'er, Ausgabe 5, Seite 112 in Zeilennummer 50000 das Zeichen »^« und in Ausgabe 7 auf Seite 70 (Supervoc) in Zeile 420 IF A$=»T« das reverse T und wie gibt man dieses Zeichen ein? Gibt es einen Ersatz dafür?

Stefan Rolfes

Leider haben wir unseren Epson FX80 noch nicht dazu gebracht, sämtliche Zeichen des Commodore zu drucken. Vor allem bei dem Pfeil nach oben und Pfeil nach links weigert er sich bisher, sie richtig auf das Papier zu bringen. Der »^« bedeutet also Pfeil nach oben.

Das oben erwähnte reverse T steht für die DEL-Taste. Man kann das Zeichen so eingeben: """(DEL)(INST) (DEL)". Also zuerst zwei Anführungszeichen, danach die DELete-Taste, die Insert(INST)-Taste, wiederum die DELete-Taste und zum Schluß noch einmal ein Anführungszeichen. Allerdings kann man die DEL-Taste auch so abfragen: IF A$=CHR$(20) THEN...

Fragen Sie doch!

Selbst bei sorgfältiger Lektüre von Handbüchern und Programmbeschreibungen bleiben beim Anwender immer wieder Fragen offen. Viel mehr Fragen ergeben sich bei Computer-Interessenten, die noch keine festen Kontakte zu Händlern, Herstellern oder Computerclubs haben. Sie können der Redaktion Ihre Fragen schreiben oder Probleme schildern (am einfachsten auf der beigehefteten Karte). Wir veranlassen, daß die Fragen von einem Fachmann beantwortet werden. Allgemein interessierende Fragen und Antworten werden veröffentlicht.

Programme direkt vom Fernseher?

Wie kann man die Datasette VC 1530 für direkte Aufnahme von Computer-Programmen vom Fernseher (Video-Recorder) umrüsten (siehe Computer-Club-Sendungen und so weiter)?

Wolfgang Schulte

Daten speichern ohne Datei zu laden?

Wie kann ich Daten auf Diskette speichern, ohne die Datei zu laden, umzuändern und neu zu speichern (das ist mir zu umständlich)?

The Dynamike, Emden

Mit einer relativen Datei ist das ohne Schwierigkeiten möglich. In Ausgabe 6/84 wurde sie ausführlich beschrieben. Und wem diese Methode noch zu umständlich erscheint, weil ja noch das Programm geladen werden muß, kann sich dieses Programm ja auf EPROM brennen. Dann genügt ein Einstecken des Moduls in den Expansionsport, Computer einschalten und schon können die ersten Daten am Bildschirm bearbeitet werden.

Rechenungenauigkeiten beim C 64?

Warum gibt der C 64 als Ergebnis der Aufgabe PRINT INT(3/0.03) nicht 100, sondern 99 an? Das gilt auch, wenn man schreibt A%=INT(3/0.03) und dann PRINT A%.

Oliver Treiber

Hardcopies mit MPS 802 und Simons-Basic?

Wir haben den C 64 und den Drucker MPS 802 (Nachfolgegerät vom VC 1526), dazu Simons Basic. Ist es möglich, im Hires- beziehungsweise Lowcol-Modus eine Kopie des Bildschirms zu erstellen?

Gerhard Schief

Das könnten Sie ja eigentlich selbst ausprobieren. Aber wenn Sie mit dem Befehl HRDCPY keine Hardcopy des normalauflösenden Bildschirms erhalten sollten, könnte der Drucker defekt sein. Eine Hardcopy des hochauflösenden Grafikbildschirms hingegen ist mit dem Simons-Basicbefehl COPY nicht möglich. Das gilt auch für den weitgehend baugleichen VC 1526.

Apple Joystick an C 64?

Gibt es ein Interface, das es ermöglicht, Apple-Joysticks an den Commodore 64 anschließen zu können? Alle Spiele, die für die Ports 1 und 2 geschrieben worden sind, sollten steuerbar sein.

Frank Zündorff

Erhöhte Lebensdauer bei Dauerbetrieb?

Wie lange kann man den C 64 laufen lassen, ohne daß er Schaden nimmt? *Burchard Laasch*

Es gibt Computer-Benutzer, die ihren Computer niemals abschalten. Er läuft dort Jahrelang, ohne das deswegen ein Defekt auftritt. Sie begründen das damit, daß bei jedem Ein- und Ausschaltvorgang die elektronischen Bauelemente warmwerden und wieder abkühlen. Da in vielen Bauelementen verschiedene Materialien verarbeitet werden, die sich bei einer Temperaturänderung unterschiedlich ausdehnen, können Spannungen auftreten. Diese andauernden thermischen Belastungen könnten die Lebensdauer vor allem der empfindlichen Halbleiter herabsetzen. Bei einem Dauerbetrieb treten diese Belastungen nicht auf. Wenn für ausreichende Belüftung gesorgt wird und der Computer nicht noch zusätzlich auf einer Heizung abgestellt wird und vor direkter Sonneneinstrahlung geschützt ist, dürften keine Probleme entstehen.

Darf man überhaupt eine Diskette in das Laufwerk legen?

Als wir uns mit dem deutschen und dem englischen Handbuch zur 1541 befaßten, stießen wir auf einige Widersprüche in bezug auf die Handhabung:

1. Im englischen Handbuch wird auf Seite 8 unten ausdrücklich darauf hingewiesen, daß beim Leuchten der grünen LED nie eine Diskette ins Laufwerk hineingeschoben oder aus ihm hinausgenommen werden darf. Im deutschen Gegenstück ist jedoch auf Seite 5 Mitte zu lesen, daß auch im ausgeschalteten Zustand der Floppy keine Diskette darin sein darf.

2. In der deutschen Fassung des Handbuches wird gesagt, daß erst der Computer, dann das Laufwerk eingeschaltet werden soll. Die Autoren des englischen Handbuches vermerken ausdrücklich auf Seite 7, daß man nie den Computer zuerst anschalten darf, da sonst nicht »everything OK« ist. Deshalb unsere Fragen:

a) Darf man überhaupt eine Disk ins Laufwerk legen?

b) Bei welcher Reihenfolge des Einschaltens ist denn »everything OK«?

Zusammenfassend: Welches der Handbücher ist maßgeblich oder kann man beide nicht ernst nehmen?

Gregor Fromme, Justus de Zeeuw

VC 20/C 64-Tips

Manche Leserfrage zeigt deutlich, daß man doch von Zeit zu Zeit einen Blick in das mitgelieferte Handbuch werfen sollte. Auch wenn dieses Handbuch beim VC 20 nicht eben reichhaltige Informationen liefert, kann man doch auf der Seite 134 nachlesen, was Poke 36879,25 bedeutet. Mit diesem Befehl wird sowohl der Bildschirmrahmen als auch der -hintergrund auf die Farbe Weiß umgestellt. Selbstverständlich laufen Programme auch bei anderen Farbkombinationen (beim Einschalten Weiß/Hellblau) genauso gut. Wenn der Bildschirm bei R. Morgenthaler bei weißem Hintergrund ins Flackern gerät, so liegt das entweder am verwendeten Fernseher/Monitor, dann sollten Bildhelligkeit oder Kontrast etwas heruntergedreht werden. Ansonsten kann der Fehler nur im HF-Modulator liegen. Der Modulator muß dann vom Commodore-Systemhändler neu an den Computer angepaßt werden.

Wenn das Gerät auf dem grauen Markt gekauft wurde, dürfte sich das schwieriger gestalten. Ein Computer ist eben kein Gebrauchtwagen, der in jeder Hinterhofwerkstatt getestet werden kann. Hier hilft nur eines: Auf dem schnellsten Weg zum Fachhändler zu gehen und zu hoffen, daß der auch nicht bei ihm gekaufte Geräte einstellt (Selbstversuche rächen sich gerade beim HF-Modulator). Ebenso schnell sollte R. Morgenthaler das Handbuch zum VC 20 oder ein anderes Begleitbuch (die von Data Becker sind hilfreich, das von Hofacker weniger) anschaffen.

Die beiden anderen Fragen kann man zusammen beantworten. Der VC 20 läßt sich an einem gewöhnlichen Farb- oder S/W-Fernseher ohne Scart-Buchse nur mit dem HF-Modulator betreiben. Für den Betrieb mit einem Monitor oder einen Fernseher mit Spezial-(Scart)Anschluß braucht man ein Monitor-Kabel, das es für den Commodore-Farbmonitor bei jedem Systemhändler gibt. Da ich den Nordmende-Anschluß nicht kenne, kann ich nur den Rat geben, bei einem der Hardware-Drittanbieter (Data Becker oder Kingsoft etc.) oder bei Commodore Deutschland (Lyoner Str. 38, 6000 Frankfurt/M 71) anzufragen. Dort wird man gerne weiterhelfen. Ich bin gerne bereit, anderen VC 20-Anwendern — soweit ich kann — weiterzuhelfen. Meine Adresse: Adenauerallee 19, 5300 Bonn 1. Wer mir schreiben will, muß allerdings einen frankierten Rückumschlag beilegen.

Hans Altmeyer

Zur Frage von Günther Kyora: Der Anschluß von Commodore VC 20 und C64 an Fernseh-Empfänger mit Scartbuchse oder Videobuchse nach DIN ist nach folgendem Schaltplan vorzunehmen: VC 20:

Zu der punktierten Leitung: Manche Fernseh-Geräte haben einen Schalter zum Umschalten auf Monitorbetrieb (siehe Bedienanleitung). In diesem Fall kann die Verbindung entfallen. Andernfalls ist eine Schaltspannung von zirka 12 V zum Umschalten erforderlich. Dazu ist eine kleine Änderung im Computer erforderlich.

Am VC 20: Durch Umlöten von zwei Lötbrücken (E1-E3 unterbrechen, E1-E2 verbinden) wird Pin 1 des Video-Audio-Ports mit zirka 10 V belegt (vorher 5 V). Der Modulator kann nun nicht mehr ohne Änderung benutzt werden (5 V Regulator einbauen).

Am Commodore 64: Bei neueren Geräten ist eine 8polige DIN-Buchse als Video-Audio-Port eingebaut. Pin 6 bis 8 sind nicht belegt. Hier kann ein freier Anschluß (zum Beispiel 7) mit dem Ausgang des 12-V-Regulators Vr 1 des Computers verbunden werden. Bei Geräten mit 5poliger Buchse muß die Leitung direkt herausgeführt werden.

Manfred Lösch

VC 20:

Computerstecker		Videostecker	Scartstecker
5	FBAS	2	20
3	Ton	4	6
2	GND	3	17+4
1	Schaltsp.	1	8
z.B. Mas50S		Mas60	

C 64:

4	FBAS	
3	Ton	
2	GND	Wie oben Stecker
(7)	Schaltsp.	Mas80S für neue Geräte!

Starten und Speichern von Spielprogrammen beim Commodore 64

Wenn ich ein Spielprogramm von Kassette geladen habe, starte ich es normalerweise durch die Eingabe des Wortes RUN nebst Drücken der RETURN-Taste. In einigen Fällen funktioniert das nicht, aber ich habe bisher folgende Verfahren kennengelernt:
1. Eingabe von SYS mit Adreßzahl, wobei die Adreßzahl im meist nur einzeiligen LIST-Ausdruck steht.
2. Eingabe von SYS 64738 und dann erst SYS mit der Adreßzahl aus dem Listing.
3. Eingabe von SYS mit Adreßzahl, die nicht im Listing steht (wie kann ich diese finden, wenn ich sie nicht kenne oder vergessen habe?)
4. Löschen der Programmzeilen mit SYS-Angaben.
5. Drücken der RUN/STOP- und RESTORE-Taste.

Mich interessiert, wie diese Startroutinen zu erklären sind und ob sie bei einem neuen Abspeichern des Programmes zu eliminieren sind. Häufig kann man aus einem Spielprogramm nicht in den Normalzustand zurückkehren. Mir bleibt dann nichts anderes übrig, als den Commodore ab- und wieder einzuschalten. In manchen Fällen erscheint nach Drücken von RUN/STOP und RESTORE auf leerem Bildschirm das Wort READY und darunter der blinkende Cursor, aber nach Eingabe von Befehlen wie LOAD oder RUN wird nur die Eingabe gelöscht, und der Cursor springt zurück. Was bedeutet das? Wie kann ich hier den Commodore wieder zu sinnvoller Reaktion veranlassen?

Manchmal passiert mir folgendes: Ich möchte ein Spielprogramm von Kassette auf einer anderen Kassette abspeichern. Obwohl das Programm mit Namen auf der Kassette ist, kann ich es mit diesem Namen nicht saven. Der Commodore zeigt einen Error an. Lasse ich den Namen weg, funktioniert das Abspeichern einwandfrei. Wie ist das zu erklären und wie kann ich in solchen Fällen den Namen doch mitspeichern?

Werner Kiefert

Zu den Fragen zum Starten von Programmen:
1. Hier handelt es sich um ein Maschinensprache-Programm, an das vorher eine Basic-Zeile gehängt wurde, die mit dem entsprechenden SYS-Befehl den Maschinenspracheteil startet. (Der Speicher des C 64 ist aufgeteilt in die Speicherzellen 0 bis 65535. In einer dieser Speicherzellen liegt der Beginn des Maschinenprogramms. Mit dem SYS-Befehl und der Angabe der Speicherzelle, wo das Maschinenprogramm beginnt, wird es gestartet.)
2. Wenn man erst mit SYS 64738 den Computer in den Einschaltzustand bringt und dann den entsprechenden SYS-Befehl aus dem Listing eingibt, müßte eigentlich das gleiche herauskommen, wie wenn man den SYS 64738 wegläßt. Hier hat Ihnen jemand Humbug erzählt oder Ihnen ein ziemlich verpfuschtes Programm gegeben.
3. Hier wurde ein Maschinenspracheprogramm abgespeichert, ohne eine Basic-Zeile mit dem entsprechenden SYS-Befehl vornedranzuhängen. Die Adresse für den Start des Programms läßt sich am besten so finden:
— Maschinensprache lernen
— Maschinensprache-Monitor laden
— SUCHEN
4. Ist uns nicht bekannt.
5. Der sogenannte Restore-Vektor wurde vom Programmierer geändert, das heißt bei Drücken von RUN/STOP-RESTORE wird nicht mehr die normale Betriebssystem-Routine gestartet, sondern es wird zu der Adresse gesprungen, die der Programmierer angewählt hat, in diesem Fall die Startadresse des Programms.

Zur Frage, die das Zurückkehren aus Spielprogrammen in den Normalmodus betrifft:
Das war offensichtlich Absicht der Programmierer der Spiele (aus welchem Grund auch immer). Hier hilft meistens wirklich nur das Aus- und Anschalten des Computers.

Antwort auf die Frage, die das Speichern von Programmen betrifft:

Hier reicht offensichtlich der Speicher nicht mehr aus, um den Namen dort abzulegen. (Der Fehler ist dann wohl ein OUT OF MEMORY ERROR.) Man kann sich jedoch manchmal abhelfen, indem man den Variablenspeicher in einen freien Speicherbereich verlegt:

```
LOAD »Programm«
POKE 56,207
CLR
SAVE»Name«
```

Für das Funktionieren dieser Maßnahme kann nicht garantiert werden, da es doch immer sehr auf das einzelne Programm ankommt.

(gk)

Probleme mit Speichererweiterung

Frage: 3-KByte-Spiele funktionieren nicht mit dem 16-KByte-Modul? *Daniel Hüller*
Ausgabe: 7/84

Programme, die für eine 3-KByte-Erweiterung gedacht sind, laufen auf dem VC 20, wenn man bei eingesteckter 16-KByte- und 3-KByte-Erweiterung (Modulbox) folgende Zeile im Direktmodus eingibt: POKE 44,4: POKE 1024,0: POKE 56,30: POKE 648,30: NEW (RETURN). Nach Drücken der RETURN-Taste verschwindet der Cursor. Jetzt sind RUN/STOP und RESTORE zu drücken. Durch Eingabe von SYS 58238 erscheint das Anfangsbild und die Meldung: 6655 Bytes Free. So spart man sich das lästige Umstecken der Erweiterungen.

Oliver Eichhorn

»Alte« 64'er-Ausgaben

Bitte schicken Sie mir (gegen Nachnahme) Kopien des Kurses Grafikgrundlagen Teil 1 und Teil 2, da ich die entsprechenden 64'er-Ausgaben nicht mehr bekommen kann.
Martin Tobrock

Solche oder ähnliche Fragen werden nicht selten gestellt. Leider können diese Fragen nur noch negativ beantwortet werden. Alle Ausgaben 4, 5 und 6 sind mittlerweile ausverkauft! Aber wir geben auch ein Trostpflaster: Gegen Ende des Jahres werden wir eine Sonderausgabe des 64'er herausbringen, in der unter anderem sämtliche bis dahin abgeschlossenen Kurse noch einmal veröffentlicht werden.

Ohne Kommentar

Ich habe da ein Programm, das heißt »Digi-Uhr«. Es macht aus dem C 64 nichts weiter als eine Digitaluhr, deren Zeitanzeige dann auf dem Bildschirm erscheint. Ich habe es geschenkt bekommen und viel mehr ist es wohl nicht wert. Es ist schade um die Mühe, die der unbekannte Programmierer sich damit gemacht hat. Denn das ist doch ein schlechter Witz: Eine Digitaluhr, deren Ganggenauigkeit sehr zu bezweifeln ist, für sage und schreibe 1750 Mark (Rechner plus Floppy plus Fernseher). Eine drahtlose Nebenstelle der Braunschweiger Atomuhr bekommt man als Bausatz für 400 Mark. Und nun kommt in Ihrer Zeitung der zweite Witz dieser Art. — Das Blumengießen mit dem Commodore. Also wissen Sie, mit einer Billigschaltuhr und einem Verzögerungsrelais wäre es nämlich auch gegangen und dann auch noch so einfach, daß es jeder halbwegs begabte Schuljunge zusammenlöten kann!

Kommen wir nun zum Thema Spiele und als Aufhänger dazu diene uns der Q-Bernd. Da haben wir wieder mal eines der unzähligen Spiele, bei denen es mir einfach unbegreiflich ist, was deren Urheber wie Konsumenten eigentlich daran finden. Von vornherein steht fest, daß kein Erfolg möglich ist. Q-Bernd und mit ihm der Spieler sind auf der Verliererstraße.

Man verlange doch mal von einem gewöhnlichen Menschen, er solle auf einem Drahtseil einen Teich überqueren. Man sage ihm weiter, daß er nicht nur kein Geld dafür bekommt, sondern stattdessen auch noch bezahlen muß. Und man sage ihm weiter, daß man alles tun wird, um ihn da runterzuschmeißen, zum Beispiel das Seil mit Seife einschmieren, Windmaschinen aufzustellen und dergleichen. Er wird sich an die Stirn tippen und recht hat er. Aber man gebe ihm ein Computerspiel, bei dem er ebenfalls absolut keine Chance für ein Erfolgserlebnis hat und er wird zahlen und spielen und spielen und spielen — das begreife ich nicht. Mir hat man mal ein Spiel namens »Snokie« angeboten. Als der arme Vogel zum fünften Male auf die Schnauze fiel und starb, habe ich mir an die Stirn getippt und den Verkäufer gefragt, ob er mich für blöd hält. Genauso bescheuert finde ich Pacman, der kaum über die erste, aber ganz bestimmt nicht über die zweite Runde kommt. Und falls es doch ein Computerfreak schafft, sollte es mich nicht wundern, wenn der elektronische Heini dann eine Spielstufe zulegt und es ist wieder Essig. Ich habe sogar mal ein Programm erlebt, in dem der Computer jedesmal, wenn er den Spieler ausgetrickst hatte, auch noch höhnisch auf dem Bildschirm triumphierte »Got You!« In die Mentalität derer, die sowas machen, kann ich mich noch hineindenken: Sie treten gegen die Spieler an und wollen sich von diesen nicht schlagen lassen. Nur vergessen sie, daß sie sich dabei eines gewaltigen Verbündeten bedienen. Dem User bleibt so keine Chance.

Was aber sind das für Leute, die sowas benutzen? Ein normaler Mensch strebt doch nach Erfolgserlebnissen? Wer kauft sich eigentlich Mißerfolgserlebnisse oder tippt sie mühsam ein? Wie erfreulich ist doch dagegen so ein Spiel wie »Jawbreaker«, wenn man es mal wieder geschafft hat und es kommt die Zahnbürste! Und warum gibt es fast keine Spiele, bei denen zwei oder mehr Spieler gegeneinander auf dem Computer spielen, statt abwechselnd gegen den Computer, wo sie dann abwechselnd verlieren? Es war doch immer der Reiz des Spieles, zu gewinnen oder zu verlieren. Aber gegen einen Computer? Der dämliche Kerl ärgert sich doch nicht mal, wenn er wirklich verliert! Gerade das aber sollte man den Spielemachern als Tip geben: Der Computer muß auch von Anfängern zu besiegen sein und wenn man es mal geschafft hat, muß eine Printzeile ins Programm: »Scheiße! Komm du mir noch mal ans Keyboard!«

Nun zu dem Programm »Programm-Verwaltung«. Unter dem Namen kann man es natürlich nicht abspeichern, der ist zu lang. Wie wäre es mit »Filister«? Auf jeden Fall sollten Sie auf etwas hinweisen: In den Zeilen 2150 und 1520 findet man jedesmal einen Paragraphen §, der auf der C 64-Tastatur gar nicht vorkommt und den ich auch nur dank Beckers Textautomat jetzt schreiben kann. Mir ist erst nach geraumer Grübelei aufgegangen, daß er dort stellvertretend für den Klammeraffen steht.

Zu dem Programm hätte ich aber noch einiges zu sagen. Zunächst einmal grundsätzlich: Ich verstehe leider bitterwenig vom Programmieren und vieles in dem Programm ist mir glattes Chinesisch. Wenn ich jetzt Wünsche oder Verbesserungsvorschläge habe, so sei damit nichts gegen den Autor gesagt, denn ich weiß nicht, ob das, was ich gern hätte, machbar ist und ob es schwierig ist. Ein solches Programm fehlt mir schon lange, aber: Man müßte eingeben können, auf welcher Disk diese Files stehen, ob auf 12A oder 7B. Das wäre gewaltig, so aber muß man immer noch die Directory-Ausdrucke durchgeben, wenn man mal eins sucht.

Weiter: Der C 64 ist damit fürchterlich langsam. Um 165 Files alphabetisch zu sortieren, hat er etwa eine Stunde lang gebrütet. Das hätte ich nun selber schneller gekonnt. Da wünschte ich mir, daß der Computer sich akustisch meldet, daß er piept oder hupt, wenn er fertig ist oder pfeift wie ein Teekessel.

Heinrich Carstensen

Wollen Sie antworten?

Wir veröffentlichen auf dieser Seite auch Fragen, die sich nicht ohne weiteres anhand eines gutes Archivs oder aufgrund der Sachkunde eines Herstellers beziehungsweise Programmierers beantworten lassen. Das ist vor allem der Fall, wenn es um bestimmte Erfahrungen geht oder um die Suche nach speziellen Programmen beziehungsweise Produkten. Wenn Sie eine Antwort auf eine hier veröffentlichte Frage wissen — oder eine andere, bessere Antwort als die hier gelesene — dann schreiben Sie uns doch. Die Antworten werden wir in einer der nächsten Ausgaben publizieren. Bei Bedarf stellen wir auch den Kontakt zwischen Lesern her.

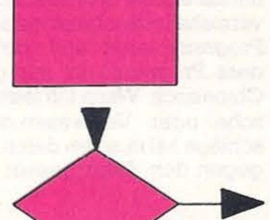
Tips für den Umgang mit Sinnbild

Für jemanden, der noch nichts mit Programmablaufplänen zu tun hatte, ist es meist schwer, seine Ideen und Programme in Form von Programmablaufplänen darzustellen. Deshalb nun einige Tips, für den Umgang mit Programmablaufplänen.

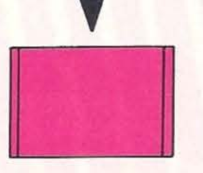
1. Operation: Dieses Sinnbild wird für Berechnungen, Zuweisungen und so weiter verwendet, ausgenommen für diejenigen Operationen, für die ein spezielles Sinnbild vorhanden ist.

2. Verzweigung (Entscheidung): Das Sinnbild hat einen Eingang und zwei Ausgänge. Je nachdem ob die Bedingung, die im Sinnbild angegeben ist, erfüllt ist, wird der Programmlauf bei dem einen oder anderen Ausgang fortgesetzt. Man sollte die Ausgänge mit ja oder nein (+ oder —) kennzeichnen, um zu verdeutlichen, welcher Ausgang benutzt wird, wenn die Bedingung erfüllt ist oder nicht.

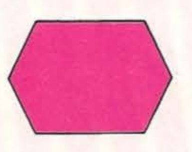
Tabelle 1: Sinnbilder für Programmablaufpläne (nach DIN 66001)



OPERATION, allgemein
Insbesondere für Operationen die im folgenden nicht besonders aufgeführt sind.
z.B.: Berechnungen, Zuweisungen, Dimensionierungen, usw.



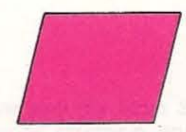
VERZWEIGUNG
Ein Sonderfall der Verzweigung ist der programmierte Schalter.



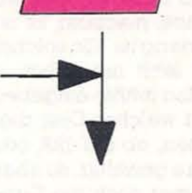
UNTERPROGRAMM
Ein Unterprogramm muß nur einmal näher dargestellt werden. Im PAP genügt dann der Hinweis wann welches Unterprogramm eingesetzt wird.



PROGRAMMODIFIKATION
z.B. das Stellen von programmierten Schaltern oder das Ändern von Indexregistern.



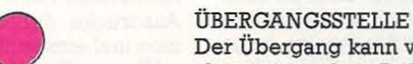
OPERATION VON HAND
z.B. Formularwechsel, Bandwechsel, Eingriff des Bedieners bei der Prozeßsteuerung.



EINGABE, AUSGABE
Ob es sich um eine maschinelle oder manuelle Ein- oder Ausgabe handelt, soll aus der Beschriftung hervorgehen.

ABLAUFLINIE, ZUSAMMENFÜHRUNG
Vorzugsrichtungen sind: von oben nach unten, und von links nach rechts. Abweichungen können durch Pfeile deutlich gemacht werden. Zwei sich kreuzende Ablauflinien bedeuten keine Zusammenführung.

ÜBERGANGSSTELLE
Der Übergang kann von mehreren Stellen aus, aber nur zu einer Stelle hin erfolgen.



GRENZSTELLE
Für A kann z.B. Beginn, Ende, Zwischenhalt eingeschrieben werden.

BEMERKUNG
Dieses Sinnbild kann an jedes Sinnbild dieser Norm angefügt werden.

3. Unterprogramm: Häufig benutzte Programmteile, die an verschiedenen Stellen des Programms benötigt werden, führt man meist als Unterprogramme aus. Im Programmablaufplan wird immer dann das Zeichen für ein bestimmtes Unterprogramm eingesetzt, wenn es benötigt wird. Wie Unterprogramme im einzelnen aussehen braucht man nur einmal gesondert aufzuführen.

4. Programmodifikation: In vielen Programmen taucht das Bedürfnis auf, bestimmte Veränderungen an Indexregistern vorzunehmen, zum Beispiel: Abschalten der Tastatur oder Wahl eines anderen Zeichensatzes. Diese Veränderungen werden in Basic meist durch Poke-Befehle bewirkt. Am Sinnbild sollte deshalb ein sehr ausführlicher Text angebracht sein, aus dem deutlich hervorgeht, was ein derartiger Eingriff im Einzelnen bewirkt.

5. Operation von Hand: Bei manchen Programmen ist es notwendig, daß der Bediener persönlich eingreift. Am Sinnbild sollte deshalb genau erläutert werden, welche Tätigkeiten vom Bediener ausgeführt werden müssen.

6. Eingabe, Ausgabe: Aus der Beschriftung des Sinnbildes sollte außerdem deutlich hervorgehen, was durch wen und welche Geräte ein- beziehungsweise ausgegeben werden soll.

7. Ablauflinie: Die Ablauflinie ist ein einfacher durchgehender Strich, der die Sinnbilder verbindet. Wenn es nicht ganz deutlich ist, wie die Ablaufrichtung verläuft, sollte sie durch Pfeile markiert werden.

Zusammenführung: Eine Ablauflinie, die in eine andere Ablauflinie einmündet, nennt man Zusammenführung. Um die Ablaufrichtung deutlich zu machen, empfiehlt es sich, Pfeile anzubringen.

ern von Programmablaufplänen

8. Übergangsstelle: Damit es nicht zu einem chaotischen Wirrwarr an Ablauflinien kommt, empfiehlt es sich, Übergangsstellen einzusetzen. Zusammengehörende Übergangsstellen müssen die gleiche Bezeichnung tragen (siehe auch Beispiel 2).
9. Grenzstelle: Mit diesem Zeichen bezeichnet man den Beginn oder das Ende von Programmen und Unterprogrammen. Es wird damit deutlich gemacht, wo ein bestimmter Ablauf beginnt oder endet.
10. Bemerkung: Mit diesem Sinnbild sollte man nicht zu sparsam umgehen. Man kann damit jedem Sinnbild beliebig viele zusätzliche Informationen anfügen.

(B. Appel/rg)

Start

Ausgabe Text — Ausgabe auf dem Bildschirm: Dieses Programm dient zur Berechnung der Wurzel von Zahlen, für die gilt: 0 < X < = 100 Geben Sie bitte Ihre Zahl ein.

2

Eingabe der Zahl X — Eingabe der Zahl X über Tastatur und Zuordnung zur Variablen X

X > 0 ? — nein → 1 — Falls X < = 0 weiter bei 1

X < = 100 ? — nein → 1 — Falls X > 100 weiter bei 1

Berechnung Y = √ X — Berechnung der Wurzel aus X Zuordnung des Ergebnisses der Variablen Y

Ausgabe Ergebnis — Ausgabe des Ergebnisses auf den BS Die Wurzel aus X = Y

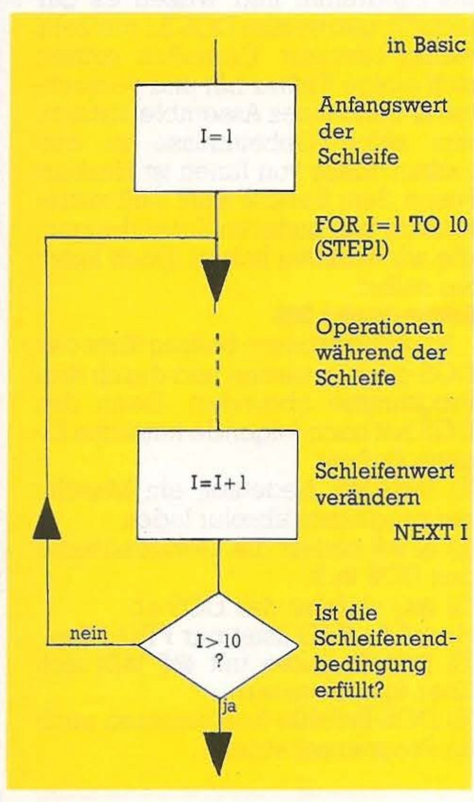
Ende

1

Ausgabe Fehlermeldung — Ausgabe auf dem Bildschirm Die eingegebene Zahl liegt außerhalb des zulässigen Bereichs. Bitte geben Sie neu ein.

2 — weiter bei 2

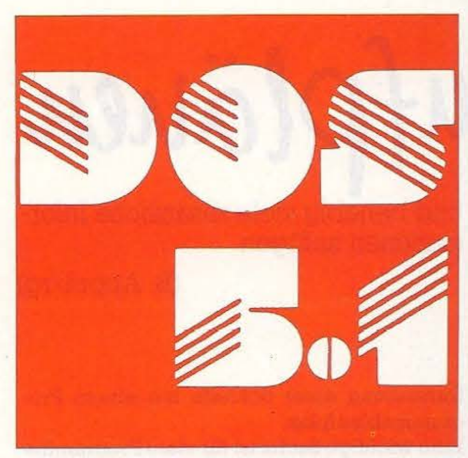
Darstellung einer Schleife mit einem Programmablaufplan
Eine wichtige Sache ist für viele Programme das Programmieren von Schleifen. Wie man eine Schleife aus den Sinnbildern von Tabelle 1 aufbaut, zeigt folgendes Beispiel.



Listing

```
20 PRINT" Dieses Programm dient zur
Berechnung der Wurzel aus einer Zahl,
für die gilt
0 x 100
Geben Sie bitte ein:"
30 INPUT X
40 IF X 0 THEN GOTO 70
50 IF X 100 THEN GOTO 70
60 GOTO 100
70 PRINT" Die eingegebene Zahl liegt
außerhalb des zulässigen Bereichs.
Geben Sie bitte neu ein."
80 GOTO 30
90 REM Berechnung
100 Y = SQR (X)
110 REM Ergebnis
120 PRINT "Die Wurzel aus"
130 PRINT X
140 PRINT "ist gleich"
150 PRINT Y
160 END
```

Das interessante Programm DOS 5.1 auf Ihrer Demo-Diskette bietet noch mehr als wir bisher gedacht hatten. Einige Leser schrieben uns: Sie entlockten dem DOS noch einige bemerkenswerte Geheimnisse. Außerdem wird das Programm noch um einige Befehle erweitert, zum Beispiel durch Merge.

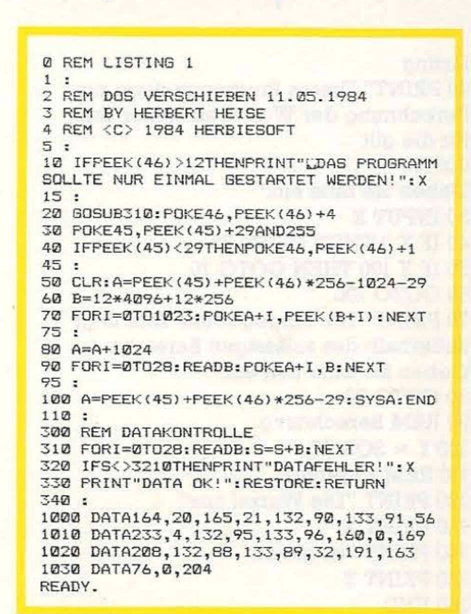


Wie schon in der Ausgabe 5/84 des 64'er-Magazins berichtet (»Wie bitte, Sie besitzen ein tolles Programm und wissen es gar nicht?«), gab es zum DOS 5.1 nur sehr wenig Literatur. Daraufhin setzten sich einige Tüftler hin und versuchten anhand eines Assemblerlistings, ihm seine Geheimnisse zu entlocken. Einer von Ihnen ist Herbert Heise. Sein Bericht steht stellvertretend für alle anderen Zusendungen, die wir erhalten haben. Doch lesen Sie selbst.

Geheimnisvolles DOS

Ich würde diesen Beitrag über das DOS gern fortsetzen und durch drei Programme abrunden. Denn das DOS hat noch folgende nützliche Eigenschaften:

(1) %NAME bedeutet, ein Maschinenprogramm absolut laden.
(2) @#9 ändert die Geräteadresse des DOS in 9.
(3) @Q schaltet das DOS ab.
(4) @$:*=PRG listet nur PRG-Files.
(5) Das Auslisten mit @$ läßt sich über Space steuern.
(6) DOS-Befehle funktionieren auch im Programmbetrieb.



```
0 REM LISTING 1
1 :
2 REM DOS VERSCHIEBEN 11.05.1984
3 REM BY HERBERT HEISE
4 REM <C> 1984 HERBIESOFT
5 :
10 IFPEEK(46)>12THENPRINT"␣DAS PROGRAMM
SOLLTE NUR EINMAL GESTARTET WERDEN!":X
15 :
20 GOSUB310:POKE46,PEEK(46)+4
30 POKE45,PEEK(45)+29AND255
40 IFPEEK(45)<29THENPOKE46,PEEK(46)+1
45 :
50 CLR:A=PEEK(45)+PEEK(46)*256-1024-29
60 B=12*4096+12*256
70 FORI=0TO1023:POKEA+I,PEEK(B+I):NEXT
75 :
80 A=A+1024
90 FORI=0TO28:READB:POKEA+I,B:NEXT
95 :
100 A=PEEK(45)+PEEK(46)*256-29:SYSA:END
110 :
300 REM DATAKONTROLLE
310 FORI=0TO28:READB:S=S+B:NEXT
320 IFS<>3210THENPRINT"DATAFEHLER!":X
330 PRINT"DATA OK!":RESTORE:RETURN
340 :
1000 DATA164,20,165,21,132,90,133,91,56
1010 DATA233,4,132,95,133,96,160,0,169
1020 DATA208,132,88,133,89,32,191,163
1030 DATA76,0,204
READY.
```

Listing 1. Mit diesem Programm wird das DOS 5.1 kopierbar wie ein normales Basicprogramm.

Als ich mir das Diskettenlaufwerk gekauft hatte, wußte ich mit dem DOS wenig anzufangen. Dies liegt wohl hauptsächlich an der mangelnden Beschreibung im Handbuch. Mit der Zeit erfuhr ich durch Zufall und durch Bekannte, daß hinter dem DOS doch mehr steckt. Ich wurde neugierig und schaute mir das DOS genauer an. Damit das besser ging, erstellte ich mir zum DOS zuerst ein Assemblerlisting. Ich habe es gelesen und versucht, es zu verstehen. Was ich dabei herausgefunden habe, möchte ich hier beschreiben.

Später werde ich ein Programm 1 vorstellen, das es vereinfacht, das DOS zu kopieren.

Ein Programm 4 wird den Namen (=Einschaltmeldung) ändern.

Zum Schluß erweitere ich das DOS um den MERGE-Befehl. Über »+« kann damit ein Programm zugeladen werden.

Ich empfehle, zuerst das DOS zu laden.

Erklärung der Befehle /,↑,%

Das DOS erlaubt, ein Programm auf drei verschiedene Arten zu laden.

Um die Beschreibung etwas zu vereinfachen, werde ich die Programme, die ich laden will, einfach »Test« nennen. Normalerweise soll ein Basicprogramm geladen werden. Über Basic gebe ich »LOAD " TEST ",8« ein. Beim DOS erspart man sich die Anführungszeichen und schreibt ganz kurz: /TEST. Soll nach dem LOADING sofort ein RUN ausgeführt werden, schreibt man einfach ↑ TEST.

```
0 REM LISTING 2
1 :
2 REM DOS-ROUTINE 15.05.1984
3 REM BY HERBERT HEISE
4 REM <C> 1984 HERBIESOFT
5 :
200 LDY $14  ; SYS-EINSPRUNGPUNKT
210 LDA $15
220 STY $5A  ; = BLOCKENDE + 1
230 STA $5B
240 SEC
250 SBC #$04 ; BLOCKENDE - 4*256
260 STY $5F  ; = BLOCKANFANG
270 STA $60
280 LDY #$00 ;
290 LDA #$D0
300 STY $58  ; = NEUES BLOCKENDE +1
310 STA $59
320 JSR $A3BF; BLOCK-VERSCHIEBE-ROUTINE
330 JMP $CC00; DOS-EINSPRUNG
READY.
```

Listing 2. Die im Listing 1 enthaltenen DATA-Werte als Assemblerlisting.

Die dritte Möglichkeit besteht darin, ein **Maschinenprogramm absolut zu laden**. Da dies seltener vorkommt, wird dies meist verschwiegen. In Basic lautet der Befehl LOAD " TEST ",8,1. Geschieht dies im Direktmodus, so teilt mir der Computer bald sein OUT OF MEMORY mit und ich muß ein NEW eingeben. Dies wird beim DOS mit %TEST vermieden. Um diesen Befehl testen zu können, wird natürlich ein Maschinenprogramm benötigt. Das DOS ist zwar ein solches, versucht man es aber mit zum Beispiel %DOS so erhält man eine

```
0 REM LISTING 3
1 :
2 REM DOS GESTUTZT
3 REM BY HERBERT HEISE
4 REM <C> 1984 HERBIESOFT
5 :
10 POKE53280,9
20 POKE53281,0
30 POKE650,128
40 PRINTCHR$(147);CHR$(158)
90 :
100 A=PEEK(45)+PEEK(46)*256-29:SYSA:END
READY.
```

Listing 3. In Verbindung mit Listing 1 lassen sich Bildschirmfarben und andere Einstellungen mühelos integrieren.

Fehlermeldung. Diejenigen, die von Data Becker »Profimat« oder »Profiass« besitzen, können dies damit probieren.

Ein Tip am Rande

Man spart sich Tipparbeit, wenn man sich mit @$ das Inhaltsverzeichnis auf den Bildschirm holt. Danach fährt man mit dem Cursor hoch zum gewünschten Programmnamen und drückt die Tasten »/«, »↑«, »%« oder »←« und ein RETURN. Schon erscheint ein LOADING beziehungsweise SAVING des Programms.

Wer der Meinung ist, das DOS läßt sich nur mit einer Floppy einsetzen, den kann ich jetzt eines Besseren

belehren. Soll das DOS zum Beispiel mit einem Kassettenrecorder arbeiten, genügt die Eingabe »@ # 1«. Dadurch adressieren wir Gerät 1. Es können nun die Abkürzungen /,↑,% benutzt werden. Natürlich kann man mit @ keinen Fehlerkanal lesen. Da nach jedem SAVE-Befehl mit »←« versucht wird, diesen zu lesen, kann auch diese Abkürzung nicht verwendet werden.

DOS 5.1 arbeitet mit Disk oder Kassette

Die Glücklichen, denen zwei Diskettenlaufwerke zur Verfügung stehen, wählen zum Beispiel über: @ # 9 ihre Station aus.

```
0 REM LISTING 4
1 :
2 REM NEUER NAME FUER DOS 15.05.1984
3 REM BY HERBERT HEISE
4 REM <C> 1984 HERBIESOFT
5 :
10 Z$=CHR$(147)
20 Z$=Z$+CHR$(17)+"          HERBIES DOS M
ANAGER"+CHR$(13)
30 Z$=Z$+CHR$(17)+"          CHECKED BY ME
"+CHR$(13)
40 Z$=Z$+CHR$(17)+"          (C) 1984 HERB
IESOFT"
90 :
100 L=LEN(Z$):P=52347
110 IFL>98THENPRINT"STRING TO LONG!!!":S
TOP
120 FORI=1TOL:A$=MID$(Z$,I,1)
130 POKEP+I,ASC(A$):NEXTI
140 POKEP+I,0:SYS52224
READY.
```

Listing 4. Verändern Sie den Namen des DOS.

Anmerkungen:

Die Geräteadresse wird beim Aufruf mit »SYS 52224« festgelegt. Das DOS nimmt sich dabei den Wert aus der Speicherstelle 186 (= $ba). Demnach wird das Gerät adressiert, von dem das DOS geladen wurde.

Sicher kennen viele den 'RESET-Befehl' über 'SYS 64738'. Damit bringt sich unser Rechnergenie in den Einschaltzustand. Soll das DOS danach wieder arbeiten, müssen wir ihm die richtige Gerätenummer zuweisen (zum Beispiel @ # 8).

Wer hätte das gedacht: Das DOS kann sich abschalten. Die Syntax lautet »@Q«. Eine Kontrolle mit dem Klammeraffen »@« ergibt SYNTAX ERROR.

Dies ist nützlich bei Programmen, die sich mit dem DOS nicht vertragen wollen. Beispielsweise tun dies Simons Basic und manche BACKUP-Programme nicht. Natürlich kann ersatzweise auch kurz der »Saft« abgedreht werden (das heißt Computer ausschalten).

DOS und das Inhaltsverzeichnis

Die Diskette hat viele Vorteile, unter anderem wird auf dieser automatisch ein Inhaltsverzeichnis angelegt. Dieses ist als Programm »$« gespeichert. Man kann es also über LOAD"$", 8 laden und mit LIST anschauen. Das DOS bietet diesen Befehl auch an. Er lautet »@$«. Dabei wird ein eventuell vorhandenes Programm nicht überschrieben. Zusätzlich kann das Auslisten gestoppt werden. Es genügt ein Druck auf die SPACE-Taste und das DOS verweilt in einer Warteschleife. Ein zweiter Tastendruck veranlaßt das DOS

```
0 REM LISTING 5
1 :
2 REM DOS MIT MERGE 15.05.1984
3 REM BY HERBERT HEISE
4 REM <C> 1984 HERBIESOFT
5 :
10 FOR I=0 TO 36
20 READA:S=S+A
30 POKE53081+I,A
40 NEXT
50 IF S<>4511 THEN PRINT"DATAFEHLER!":EN
D:X
60 SYS53081
70 PRINT"OK!"
80 END
90 :
1000 DATA169,043,141,032,204,162,104
1001 DATA160,207,142,021,204,140,010
1002 DATA204,096,169,047,141,122,204
1003 DATA169,255,133,020,133,021,032
1004 DATA019,166,166,095,164,096,076
1005 DATA038,206,*
READY.
```

Listing 5. Erweitern Sie das DOS mit dem Merge-Befehl.

weiterzuarbeiten. Komfortabel wird das Ganze dadurch, daß man unter den Programmnamen auswählen kann. Dies geschieht durch Angabe von »*« oder »?«. Eine dritte Möglichkeit erlaubt Filetypen zu unterscheiden. »@$:* = PRG« listet alle vorhandenen PRG-Files aus. Entsprechendes gilt für »@$:T?S = SEQ« und »@ $:TES* = USR«.

DOS innerhalb von Programmen

Aus dem Beitrag über das DOS in der Ausgabe 5/84 geht hervor, daß man das DOS nur im Direktmodus benutzen kann. Dem muß ich widersprechen. Die Befehle haben allerdings eine andere Form. Allgemein ist es notwendig, **den Namen in Anführungszeichen** zu setzen. Die Befehle lauten dann zum Beispiel »@"$"«, »/"TEST"«, »←"TEST"« und »@" "«. So darf man auch im Direktmodus vorgehen. Ich denke aber, es wird sich keiner diese Mühe machen. Die Befehle werden ganz normal durch Doppelpunkt getrennt, zum Beispiel »@"S:TES*" :@" " :@"$"«.

DOS als Programm ohne Lader

Als ich zum ersten Mal mit dem DOS Bekanntschaft machte, störte

```
0 REM LISTING 6
1 :
2 REM DOS ERWEITERUNG MERGE 21.05.1984
3 REM BY HERBERT HEISE
4 REM <C> 1984 HERBIESOFT
5 :
1000 REM KONSTANTEN
1005 :
1010 LDA #"+"
1020 STA $CC20 ; ">" IM PROGRAMM
1030 LDX #$68
1040 LDY #$CF  :REM ADRESSE-1 VON MERGE
1050 STX $CC15
1060 STY $CC0A
1070 RTS
1090 :
2000 REM MERGE
2005 :
2010 LDA #$2F
2020 STA $CC7A
2030 LDA #$FF
2040 STA $14
2050 STA $15
2060 JSR $A613 :REM ZEILE SUCHEN
2070 LDX $5F
2080 LDY $60
2090 JMP $CE26 :REM LOAD-ROUTINE
READY.
```

Listing 6. Auch hier wieder die in Listing 5 enthaltenen DATA-Werte als Assembler-Listing.

mich die Tatsache, daß es in Form zweier Programme abgespeichert war. Das DOS ließ sich außerdem ohne Kopierprogramme schlecht auf eine andere Diskette kopieren. Deshalb habe ich aus dem DOS eine Art Basicprogramm gemacht. Dieses kann ich mit LOAD laden und mit SAVE abspeichern. Wollen Sie es mir nachmachen?

Dann laden Sie bitte das DOS auf dem bisher üblichen Weg, falls es nicht ohnehin schon geschehen ist. Tippen Sie nun das Programm »DOS verschieben« ein (Listing 1 und 2). Mit einem RUN ohne Datafehler verschiebt sich das DOS direkt hinter das Programm. Dies erlaube ich nur einmal, da das Programm sonst doppelt so lang oder länger würde. Löschen Sie jetzt alle Programmzeilen außer Zeile 100. Es darf kein NEW eingegeben werden. Es können aber beliebig Zeilen hinzugefügt werden. Zum Beispiel könnte man die Bildschirmfarben festlegen und alle Tasten mit Repeatfunktion versehen (Listing 3). Damit ist meine DOS-Version fertig und kann abgespeichert werden.





DOS unter neuem Namen

Tippen Sie das Programm »Neuer Name für DOS« (Listing 4) ein. Die Bezeichnung für das eigene DOS muß der Variablen Z$ zugeordnet werden. Hierfür werden die Zeilen 0 bis 99 verwendet. Der String darf bis zu 98 Zeichen enthalten und hat dieselbe Form wie beim PRINT-Befehl (PRINT Z$;). Ein RUN schreibt diesen String in das DOS. Jedes SYS 52224 erzeugt nun diesen Namen auf dem Bildschirm. Diese DOS-Version kann natürlich in ein Basicprogramm umgewandelt werden, wie ich es oben beschrieben habe.

MERGE-Befehl für das DOS

Es wurden schon viele Methoden vorgestellt, die es ermöglichen sollten, ein Programm so zu laden, daß es an ein bestehendes angehängt wird. Ich erlaube mir nun, dem DOS diese zusätzliche Last aufzubürden.

Das Programm »DOS plus MERGE« (Listing 5 und 6) erzeugt diesen Befehl. Als Zeichen dient »+«. Ein MERGE-Befehl sieht dann aus wie »+TEST«. »+« und »/« arbeiten fast gleich. Sie unterscheiden sich nur in der Ladeadresse.

Ich sollte noch etwas anmerken. Hat das DOS das neue Zeichen »+« akzeptiert, so wird es vergeßlich. Denn das Zeichen »>« erfüllt seine Funktion nur noch im Direktbetrieb wie »+« auch.

Wie schon in der Ausgabe 5 angedeutet, lassen sich sämtliche Befehle, die normalerweise mit OPEN 15, 8, 15,".... übertragen werden, mit dem Klammeraffen abkürzen. Dazu gehören auch die Direktzugriffsbefehle wie B-R oder etwa B-W, und so weiter. Auch der im normalen Basic mögliche Replace-Befehl: SAVE"@:name", 8 läßt sich abkürzen mit ←@:name. Damit kann man ein Programm erneut unter gleichem Namen abspeichern, ohne daß es zu einem FILE EXIST ERROR kommt. Der Replace-Befehl funktioniert natürlich auch beim erneuten Schreiben von zum Beispiel sequentielle Dateien, die bereits unter gleichem Namen existieren. Allerdings wird dabei die alte Version überschrieben. (Herbert Heise/gk)



Flußdiagramme — eine Brücke zwischen Programmierern und Programmiersprachen?

Wer hat sich als Programmierer nicht schon darüber geärgert, daß in einer Zeitschrift ein Listing eines interessanten Programms abgedruckt ist, aber ausgerechnet für einen anderen Computertyp als den eigenen? Flußdiagramme helfen, eventuelle Anpassungsprobleme auf ein Minimum zu beschränken.

Vielen Computerfreaks wird es wohl schon so ergangen sein. Man sieht in einer Zeitschrift ein interessantes Programm, aber es wurde für einen anderen als den eigenen Computer geschrieben. Also versucht man sein Glück und fängt an das Programm auf den eigenen Computer anzupassen. Meistens reicht dann die Beschreibung nicht aus, um in Verbindung mit dem Listing die nötigen Änderungen vorzunehmen. Nach einigen Stunden mühsamen Eintippens stellt man dann bedrückt fest, Basic ist nicht gleich Basic und das für den Computer XY geschriebene Programm will einfach nicht auf dem eigenen Gerät laufen. Entweder kapituliert man dann, oder man versucht solange weiter bis ein lauffähiges Programm entsteht, welches dann unter Umständen kaum noch Ähnlichkeit mit dem Original hat. Jedenfalls erreicht man dann irgendwann einmal den Punkt, wo man sich fragt, warum nicht alle Programme in einer einheitlichen Programmiersprache geschrieben werden, die von allen Computern verstanden werden können. Es ist natürlich nur mit großem Aufwand möglich, eine einheitliche Programmiersprache für alle bestehenden Computersysteme zu schreiben und außerdem wäre der Aufwand höher als der effektive Nutzen.

Es stellt sich also die Frage, ob es nicht eine Möglichkeit gibt, Programme so darzustellen, daß sie von der Hardware und der Programmiersprache weitgehend unabhängig sind. Hier stellen Flußdiagramme eine große Hilfe dar.

In der Datenverarbeitung kennt man schon lange verschiedene Möglichkeiten der Programmdarstellung. Einen besonderen Platz nimmt dabei der Ablaufplan ein. Er bietet mit seinen Sinnbildern (siehe Bild 1) die Möglichkeit ein Programm übersichtlich und detailliert darzustellen, ohne an einen bestimmten Computer gebunden zu sein.

Wie man von einer Problemstellung zu einer vollständigen Programmdokumentation kommt, soll nun an einem kurzen Beispiel erläutert werden.

Problem: Für Zahlen, die größer als 0 und kleiner oder gleich 100 sind, sollen die Quadratwurzeln berechnet werden. Um das Problem zu lösen, sollte für die einzelnen benötigten Programmteile ein Schema erstellt werden:

1. Eingabe der Zahl x
2. Überprüfung der Zahl x
3. Berechnung der Wurzel
4. Ausgabe des Ergebnisses

Ein Beispiel, das aus technischen Gründen auf Seite 15 abgedruckt werden mußte, zeigt wie das obige Schema als Programmablaufplan aussehen kann.

Anhand des ausführlichen Flußdiagramms sollte es den meisten Programmierern möglich sein, ein lauffähiges Programm zu schreiben. Die eben erwähnte Methode ist jedoch nicht die einzigste Möglichkeit einen Programmablaufplan zu dem gestellten Problem zu erstellen. In Anlehnung an die Programmiersprache Basic wurde hier die Beschriftung der Symbole geändert. Für diejenigen, die sich mit Basic auskennen, dürfte die gezeigte Version eine wesentliche Vereinfachung darstellen.

Das Beispiel zeigt, daß mehrere Möglichkeiten zur Verfügung stehen einen Programmablauf zu erstellen. Wie umfangreich die Diagramme ausfallen, hängt in erster Linie davon ab, ob man nur einen groben Überblick über das Programm vermitteln möchte, oder ob es anderen Programmierern die Möglichkeit geben soll, ein lauffähiges Programm zu erstellen.

Ein Programmablaufplan ist also, wenn genügend Informationen darin enthalten sind, eine nützliche Brücke zwischen Programmierern und Programmen. Wie stabil sie ist, hängt jedoch von einigen weiteren Faktoren ab. Eine möglichst ausführliche Programmdokumentation sollte noch Informationen enthalten, die benötigt werden, um entscheiden zu können, ob es überhaupt sinnvoll ist, das Programm auf das eigene System anzupassen.

(Burkhard Appe/rg)

CALC RESULT

DREIDIMENSIONALE KALKULATIONEN

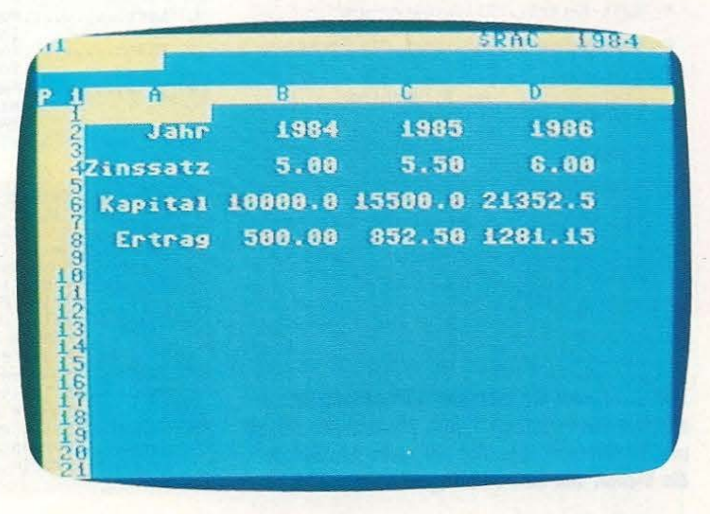
Traditionelle Kalkulationsprogramme bestehen aus einer Seite mit 64 Spalten und 254 Reihen. Calc Result hat 32 solcher Seiten, und diese sind miteinander verknüpfbar. Doch das ist nicht der einzige Grund, warum man noch einen zweiten Blick riskieren sollte. Denn dieses Tabellenkalkulationsprogramm bietet die Möglichkeit, Balkendiagramme und formatierte Ausdrucke zu erzeugen, erleichtert die Bedienung durch jederzeit einblendbare Hilfsbildschirme, schützt einmal aufgestellte Formeln, und läßt schnelle Neuberechnungen zu. Wir haben Calc Result in einem ausführlichen Test genauer unter die Lupe genommen.

Calc Result wird in vier verschiedenen Versionen angeboten. Jeweils eine für den CBM 700 und den CMB 8000 sowie zwei Versionen für den Commodore 64. Die einfachste Ausführung »Calc Result Easy« kostet 295 Mark und besteht nur aus einem Modul, das in den Modul-Steckplatz des Commodore 64 geschoben wird. Die andere Version für den Commodore 64 heißt »Calc Result Advanced« und wird mit einem Modul und einer Diskette zu einem Preis von 495 Mark angeboten. Dieselbe Konfiguration ist auch für die beiden größeren Systeme erforderlich, deshalb wollen wir hier nur »Calc Result Advanced« betrachten, zumal die Leistungsmerkmale nahezu identisch sind.

Das auffälligste Merkmal von Calc Result ist die dreidimensionale Verarbeitung von Tabellen. Was ist darunter zu verstehen? Am besten ist es, sich ein Blatt Papier vorzustellen und das Papier in 64 Spalten und 256 Reihen zu unterteilen. Auf diesem Blatt kann man nun sämtliche Berechnungen und Verknüpfungen der Zahlenkolonnen vornehmen, wie man dies von anderen Kalkulationsprogrammen gewohnt ist. Nun hat man aber nicht nur dieses eine Blatt Papier zur Verfügung, sondern ganze 32 Blätter. Die Daten und Formate sind zwischen diesen 32 Seiten vollaustauschbar. Die letzte der 32 Seiten hat einen Sonderstatus. Auf ihr können gleichartige Tabellen quasi per Tastendruck addiert werden. Ein Beispiel soll das erläutern: Sie führen monatlich eine gleichbleibende Kostenübersicht. Lediglich die Werte ändern sich von Eintragung zu Eintragung. Am Jahresende wollen Sie die Gesamtkosten wissen. Sie geben den entsprechenden Befehl ein und das Ergebnis steht auf Seite 32. Die monatlichen Kosten und das Gesamtergebnis können Sie in Diagrammform sowohl auf den Bildschirm als auch auf den Drucker ausgeben (siehe auch Bild 1). Komplizierte Berechnungen können so durch Tastendruck grafisch dargestellt werden. In das Balkendiagramm sind lediglich eine Kopf- und zwei Fußzeilen für Erklärungen nachzutragen. Alle Berechnungen werden vom Kalkulationsprogramm ausgeführt. Dies dokumentiert bereits die Bedienungsfreundlichkeit von Calc Result.

Berechnungen auf 32 Seiten

Als Kriterium für die leichte Bedienung eines Tabellenkalkulationsprogramms zählt auch die Art und Weise, wie die Befehle und Kommandos einzugeben und abzulesen sind. Neben dem Feld für die Dateneingabe sind besonders die ersten drei Zeilen von Bedeutung. Da wäre zunächst die sogenannte Befehlszeile. In dieser Zeile können Sie Befehle, den Inhalt der momentanen Cursorposition und verschie-



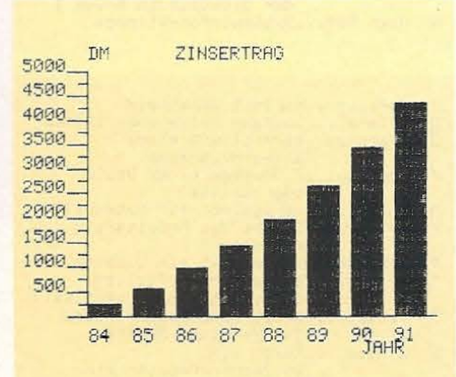


Bild 1. Komplizierte und unübersichtliche Zahlenkolonnen können grafisch dargestellt werden. Hier das Beispiel eines Zinsertrags



Bild 2. So sieht der Befehlseingabe-Modus aus

```
B: Blank.........Loescht das Feld, in
                 dem der Cursor steht
D: Disk-Befehl...Diskettenoperationen
                 und
                 Systeminformationen
E: Edit-Befehl...Bildschirm- und
                 Druckbildaufbereitung
F: Format-Befehl.Definiert das Format
                 eines Feldes
G: Global-Befehl.Definiert das Format
                 des Bildschirms
L: Leave.........Ruecksetzen von
                 Bildschirmfenstern
O: Order.........Reihenweise oder spal-
                 tenweise Berechnung
P: Page-Befehl...Seiten-Funktionen
Q: Quit..........Programm-Ende
R: Recalculate...Automatische oder ma-
                 nuelle Neu-Berechnung
-: ..............Automatische Wiederho-
                 lung eines Zeichens
```

Bild 3. Der Hilfsbildschirm mit Menü

dene Funktionen wie Angaben zum Format der momentanen Position (maximale Genauigkeit, Integer, 2 Dezimalstellen, rechts- oder linksbündig) und den freien Hauptspeicherplatz ablesen. Die zweite, die Hilfszeile, zeigt folgende vier Funktionen an: GOTO (für direkte Sprünge mit dem Cursor auf dem Arbeitsblatt), HELP (darauf kommen wir später noch zu sprechen), HARDCOPY (für einen Ausdruck des Bildschirms) und CLEAR (löscht den Bildschirm). Daneben dient diese Zeile für den Dialog. Hier wird die Eingabe abgefragt sowie auf Antworten hingewiesen. Mit Ausnahme der Druckeditierung können alle Fragen mit einem Tastendruck beantwortet werden. In der dritten Zeile, der Eingabezeile, werden die eingegebenen Zeichen sowie die Fragen, die mit mehreren Tasten zu beantworten sind, angezeigt.

Nun aber zu der Help-Funktion. Hat man vom Eingabemodus in den Befehlsmodus umgeschaltet, erscheint in der Befehlszeile folgender Ausdruck: SYSTEM: B D E F G L O P Q R – (Bild 2)

Die Help-Taste hilft weiter

Das soll übersichtlich und informativ sein? Diese Frage stellen Sie zunächst mit Recht. Drücken Sie doch einmal die Help-Taste. Es erscheint ein Hilfsbildschirm, auf dem die Auswirkung jedes einzelnen Befehls kurz erklärt wird (Bild 3). Sie erfahren, daß B für Blank steht und dadurch das Feld, in dem sich der Cursor befindet, gelöscht wird (Übrigens: Zahlenwerte in einer Zelle können einfach durch Überschreiben gelöscht werden, Formeln sind jedoch schreibgeschützt, deshalb dieser Befehl). Unter D finden Sie den Disk-Befehl mit der Erklärung: Diskettenoperationen und Systeminformationen. Sollte Ihnen das nicht ausreichen, hilft die Help-Funktion weiter. Ein zweiter Hilfsbildschirm gibt die nötigen Informationen. In Bild 4 sind die neun Unterbefehle der D-Anweisung aufgelistet. Wenn Sie sich die Erklärungen genau durchlesen, werden Sie bemerken, daß hier immer von Drive 0 und Drive 1 die Rede ist, obwohl wir in unserem Test Calc Result Advanced für den 64er verwenden, und hier das Vorhandensein von zwei Laufwerken in den seltensten Fällen realisiert sein dürfte. Im Handbuch (sehr gut, in deutscher Sprache, mit vielen Beispielen) wird zwar auf das Arbeiten mit einem Laufwerk kurz hingewiesen, aber sämtliche weiteren Erklärungen beziehen sich auf zwei Laufwerke. Hier hätte man besser et-

```
B: Backup.....Kopiert die Diskette in
              Drive 0 nach Drive 1
C: Catalog....Zeigt das Inhaltsver-
              zeichnis des Drive 1
D: DIF-file...Speichern und Laden
              einer DIF-Datei
E: Erase:.....Loescht eine Datei auf
              der Diskette in Drive 1
I: Initialize.Initialisiert Drive 0
              und Drive 1
L: Load.......Laedt eine Datei in den
              Arbeitsspeicher
N: New........Formatiert eine neue
              Diskette in Drive 1
S: Save.......Speichern der Daten auf
              der Diskette in Drive 1
U: User Reg...Systeminformationen
```

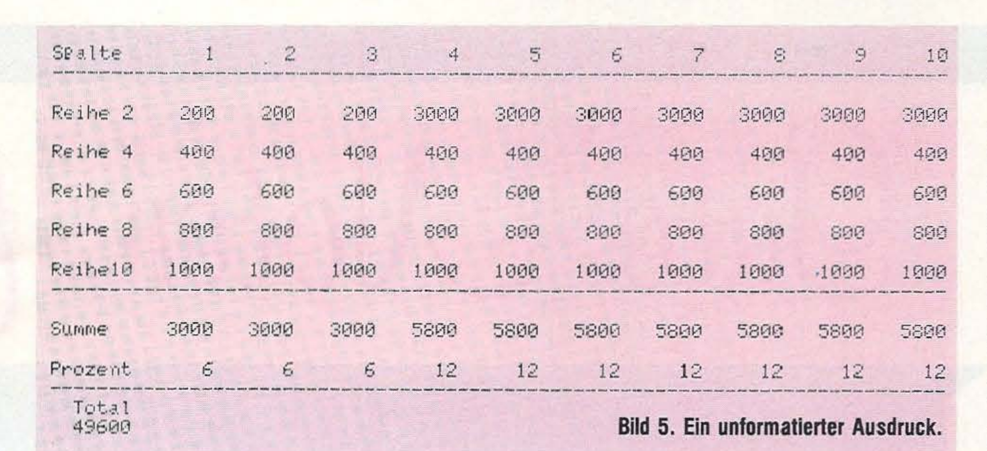
```
C: Copy......Kopiert Datenfeld
D: Delete....Loescht Zeile oder Spalte
G: Graphics..Darstellung eines
             Balkendiagramms
I: Insert....Einfuegen einer Zeile
             oder Spalte
M: Move......Verschiebt ein Datenfeld
P: Print.....Druckt das Arbeitsfeld
             formatiert
R: Replicate.Wiederholt ein Datenfeld
S: Split.....Bildschirm aufteilen
             (horizontal oder vertikal)
T: Title.....Setzt eine Beschriftung
             in die linke Spalte
W: Window....Fuegt ein
             Bildschirmfenster ein
```

```
B: Blank.........Loescht das Feld, in
                 dem der Cursor steht
D: Disk-Befehl...Diskettenoperationen
                 und
                 Systeminformationen
E: Edit-Befehl...Bildschirm- und
                 Druckbildaufbereitung
F: Format-Befehl.Definiert das Format
                 eines Feldes
G: Global-Befehl.Definiert das Format
                 des Bildschirms
L: Leave.........Ruecksetzen von
                 Bildschirmfenstern
O: Order.........Reihenweise oder spal-
                 tenweise Berechnung
P: Page-Befehl...Seiten-Funktionen
Q: Quit..........Programm-Ende
R: Recalculate...Automatische oder ma-
                 nuelle Neu-Berechnung
-: ..............Automatische Wiederho-
                 lung eines Zeichens
```

```
C: Color.........Farbauswahl
G: Global Feld...Frueheres gewaehltes
                 Globalformat
          Global.Normalformat (Label
                 nach links, Value nach
                 rechts, maximale
                 Praezision
M: Maximum.......Maximale Praezision
I: Integer.......Setzt Feld in
                 'Integer'-Format
$: ..............Rundet Feldinhalt (mit
                 2 Nachkommastellen)
L: Left..........Positioniert Feld-
                 inhalt linksbuendig
$: Right.........Positioniert Feld-
                 inhalt rechtsbuendig
*: ..............Fuellt das Feld
                 mit Sternen
                 (Anzahl = Feldinhalt)
```

```
A: Add......Seiten addieren (mit
            Label- und Formelpruefung)
C: Copy.....Seite kopieren
D: Delete...Seite loeschen vom
            Arbeitsspeicher
E: Erase....Arbeitsspeicher auf
            Drive 0 loeschen
G: Get......Seite vom Arbeitsspeicher
            (Drive 0) laden
N: Negate...Negieren alle Werte
            (values) einer Seite
P: Put......2. Seite im Arbeitsspeicher
            (Drive 0) speichern
R: Renumber.Seite neu numerieren
+: .........Seiten addieren (ohne
            Label- und Formelpruefung)
```

Bild 4. So sind die einzelnen Hilfsbildschirme miteinander verknüpft. Durch Studium der einzelnen Erklärungen läßt sich ein erster Einblick in die Vielfalt von Calc Result gewinnen.



Spalte	1	2	3	4	5	6	7	8	9	10
Reihe 2	200	200	200	3000	3000	3000	3000	3000	3000	3000
Reihe 4	400	400	400	400	400	400	400	400	400	400
Reihe 6	600	600	600	600	600	600	600	600	600	600
Reihe 8	800	800	800	800	800	800	800	800	800	800
Reihe10	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
Summe	3000	3000	3000	5800	5800	5800	5800	5800	5800	5800
Prozent	6	6	6	12	12	12	12	12	12	12
Total 49600										

Bild 5. Ein unformatierter Ausdruck.

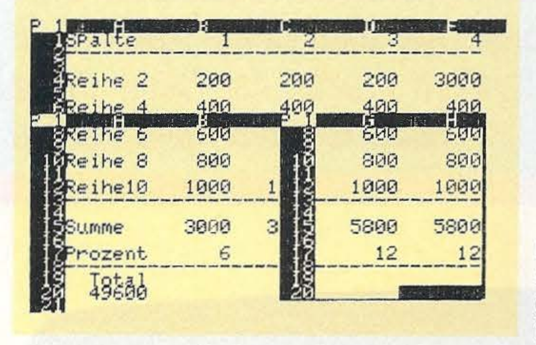


Bild 6. Ein horizontal geteilter Bildschirm mit einem Fenster, das von C7 bis E20 definiert wurde. Der Cursor (unten rechts) ist im Fenster »eingefroren«

was mehr an die Anwender gedacht, die gerne mit einem guten Tabellenkalkulationsprogramm arbeiten möchten, aber nur ein Laufwerk besitzen.

Die ersten acht Disk-Befehle dürften keine Verständnisprobleme bereiten. Der letzte Befehl U für User Register soll noch kurz erläutert werden. Mit diesem Befehl kann man die Sprache der Hilfsbildschirme bestimmen (acht verschiedene stehen zur Auswahl), die Farben und Geräteadressen einstellen sowie den Druckertyp und das Papierformat wählen. Welche Unterbefehle beim Edit-, Format- und Page-Befehl zur Verfügung stehen, kann ebenfalls Bild 4 entnommen werden.

Flexible Druckformatierung

Ein weiteres wichtiges Kriterium für die Beurteilung eines Kalkulationsprogrammes stellt für den Anwender die Flexibilität der Ausgabe seiner Berechnungen dar. Calc Result ermöglicht es, das Zahlenmaterial noch während des Ausdrucks zu bearbeiten. So kann man wählen, in welcher Reihenfolge die Spalten ausgegeben werden sollen, wie breit die Spalte sein soll und wie oft die entsprechende Spalte im Ausdruck erscheinen soll. Diese und andere Ausgabeparameter können Sie abspeichern und bei Bedarf wieder einlesen. Man kann aber, wie bei den meisten Kalkulationsprogrammen üblich, das Zahlenmaterial auch so ausgeben, wie es auf dem Bildschirm zu sehen ist (siehe Bild 5).

Da wir gerade beim Bildschirm sind, gehen wir auf dessen Darstellungsmöglichkeiten ein. Eine Bildschirmseite kann sowohl horizontal als auch vertikal geteilt werden. Die Teilung muß aber mindestens drei Reihen von der x-Achse und drei Spalten von der y-Achse entfernt sein.

Damit ergibt sich für den 64er mit seinem 40-Zeichen-Bildschirm eine maximale Dreiteilung in vertikaler Richtung und es zeigt sich unter anderem, daß es problematisch sein kann, Tabellenkalkulation mit nur 40 Zeichen pro Zeile sinnvoll zu benutzen. Den Cursor kann man beliebig in diesem geteilten Bildschirm positionieren und sich einzelne Teilbereiche durch »Abrollen des Textes« anzeigen lassen. Schwieriger wird die Sache, wenn man sich noch ein sogenanntes Bildschirmfenster definiert hat. Befindet sich der Cursor nämlich in so einem Fenster, läßt er sich durch keine noch so gut gemeinten Befehle dort wieder herausholen (siehe Bild 6). Das schränkt natürlich den intensiven Einsatz von Bildschirmfenstern ein.

Sinnvoller ist da schon die Möglichkeit, bis zu vier Seiten gleichzeitig auf dem Bildschirm darzustellen. Auf diese Weise können Sie Ergebnisvergleiche zwischen Abteilungen oder Tochtergesellschaften anstellen, Sollabweichungen errechnen und so weiter.

Zur Mathematik: Calc Result rechnet nach den Regeln der korrekten mathematischen Priorität. Es sind somit alle Rechnungsarten einschließlich Potenzrechnung möglich. Ferner stehen zur Verfügung: Mittelwert, Minimum Maximum, Standardabweichung und trigonometrische Funktionen. Auch die Verwendung von Bedienungsfunktionen wie IF THEN ELSE und OR AND NOT ist möglich. Der Formelschutz wurde bereits erwähnt. Einmal aufgestellte Formeln sind gegen versehentliches Überschreiben oder Löschen geschützt.

Daß eine Datei in jeder beliebigen Entwicklungsphase mit einer oder mehreren Seiten abgespeichert werden kann, ist selbstverständlich. Als angenehm wurde beim Laden das vorherige Auflisten der gespeicherten Dateien mit den Dateinamen empfunden. Man muß nur mit dem Cursor auf die gewünschte Datei (beziehungsweise deren Namen) fahren um diese nach zweimaliger Betätigung der Return-Taste zu laden.

Zusammenfassung

Calc Result gibt es in verschiedenen Versionen. Sie können also die Kombination wählen, die Ihnen am besten paßt. Benötigen Sie nur ein einfaches Kalkulationsprogramm, dann ist vielleicht Calc Result Easy die Lösung. Sie verzichten dabei auf die dritte Dimension, die globale Neuberechnung, die Seitenaddition und die Hilfsbildschirme. Sind die Anforderungen höher und möchten sie die oben angesprochenen Möglichkeiten nicht missen, so heißt der nächste Schritt Calc Result Advanced. Damit haben Sie ein mit einigen überzeugenden Merkmalen ausgestattetes Werkzeug für die Tabellenkalkulation zur Hand. Ihre Wahl wird auf die Versionen für den Commodore 700 oder Commodore Serie 8000 fallen, wenn Sie außer den Kalkulationsmöglichkeiten noch weitere Anforderungen (wie Verarbeitung von Visicalc-Dateien oder Kommunikationsfähigkeit) an Ihren Computer stellen. (aa)

Der angekündigte Test von Multiplan mußte aus Platzgründen in eine der nächsten Ausgaben geschoben werden. Dann wird auch PractiCalc beschrieben.

Gute Noten f

»Man müßte Klavier spielen können...«, tönte es einst Glück bei den Frauen hat man heute auch ohne Synthesizer System ersetzt der Commodore nötig ist: eine gute Portion

Der Computermusiker würde das Extended Synthesizer System ein Komposerprogramm nennen. Lieder direkt über die Tastatur einspielen, das funktioniert nicht. Man tippt sie Ton für Ton, mittels diverser Befehle ein. Dies bedeutet zwar viel Arbeit, dafür haben jedoch hier auch Theoretiker ohne Übung im Klavierspiel eine Chance, eigene Kompositionen zum Klingen zu bringen.

Das Programm wird auf Diskette zum Preis von 138 Mark geliefert.

Die Bedienungsanleitung umfaßt zirka 50 Seiten. Klar und übersichtlich aufgebaut, erklärt sie alles Wichtige zu den einzelnen Bedienungsfunktionen. Schade, daß kein eigenes Kapitel die Grundlagen der allgemeinen Musiktheorie berücksichtigt. Dies hätte sicher vielen Nichtmusikern den Umgang mit dem Programm wesentlich erleichtert. Auch die Grundlagen der Klangerzeugung kamen etwas kurz weg. Ein paar kleine Grafiken mehr, und auch der Nicht-Synthesizer-Fachmann wüßte anschließend besser Bescheid, über Wellenformen und Hüllkurvenverläufe. Sicher vermuteten die Autoren des Bedienungshandbuches, daß nur musikalisch vorgebildete Anwender mit dem Programm arbeiten. Doch das muß ja nicht immer sein. Es stimmt zwar, daß das Extended Synthesizer System in erster Linie für den musikalisch gut Vorgebildeten Nützliches bietet, doch hätte dieses Programm andererseits auch gerade für den Musikneuling großen pädagogischen Wert. Außerdem dürften sich unter den Commodore 64-Besitzern wesentlich mehr Musiklaien befinden als Musiker unter den Commodore 64-Usern. Nun denn, im Notfall hilft immer noch der Gang zum Musikalienhändler, der gerne mit Rat, Tat und Literatur zur allgemeinen Musiktheorie weiterhilft.

Nach dem Laden und Starten des Systems erscheint zunächst ein Titelbild. Mit »RETURN« aktivieren wir dann das Programm. Einige Sekunden später tauchen auf dem Bildschirm drei verschiedenfarbige Notenzeilen untereinander auf. So präsentiert sich das Extended Synthesizer System (siehe Bild) während wir mit ihm arbeiten.

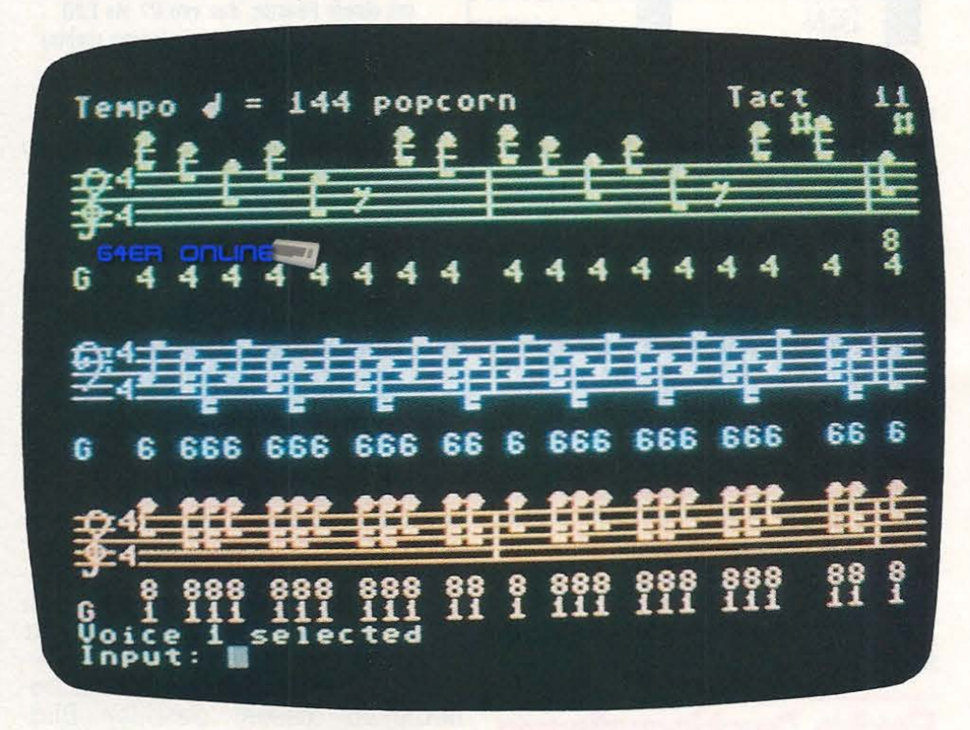


Bild 1. Der Bildschirm als Notenblatt

In der obersten Zeile erhalten wir drei Informationen zum momentan gespielten Stück. Genauer gesagt, Tempo, Name des Stücks und der gerade gespielte Takt. Die Violinschlüssel in den drei Notenzeilen darunter wirken etwas verunglückt. Doch das soll uns hier nicht weiter stören. Das Extended Synthesizer System bietet, wie es mittlerweile unter den Musikprogrammen für C 64 zum guten Ton gehört, alle drei Stimmen des SID-Chips. Es entspricht insofern einem dreifingerigen Klavierspieler. Denn genau wie dieser vermag es drei Töne gleichzeitig zum Klingen zu bringen, nicht mehr. Vorausgesetzt natürlich, die

Drei Notenzeilen am Bildschirm

Füße bleiben auf dem Pedal. Aber das sei der Fall. Eine Etage unterhalb der dritten Notenzeile, nochmals Text. Im Toneingabemodus teilt

ür gute Noten

durch die Gassen. Die Zeiten haben sich gewandelt. Fertigkeit im Klavierspiel. Denn mit dem Extended 64 die Technik des Klavierspiels. Was jedoch musikalisches Grundwissen.

uns das Extended Synthesizer System in der oberen der zwei Zeilen mit, welche der drei Stimmen wir gerade bearbeiten. Darunter eine Input-Zeile. Hier spielt sich im folgenden die gesamte Kommunikation mit dem System ab. In der Kommandozeile erscheinen sämtliche über die Tastatur eingetippten Befehle, wie auch dezente Hinweise des Systems auf etwaige Bedienungsfehler.

Spätestens jetzt ist es Zeit, einen der Demosongs zu laden. Das gibt uns den besten Eindruck von den Möglichkeiten des Systems. Töne machen die Musik.

Zunächst will das Programm jedoch immer zwei Dinge wissen: Takt und Notenschlüssel. Genau wie diese beiden Grundfesten unserer westlichen Musikkultur jede Partitur eröffnen, stehen sie auch hier am Beginn der Arbeit.

Die Eingabe des Taktes erfolgt immer in der Form a/b, mit a von 1 bis 9 und b entweder 2, 4 oder 8. Wir können also zum Beispiel 4/4, 3/4, 9/8 oder auch einen 7/8 Takt eingeben. Tippen wir aus Versehen einen unzulässigen Wert ein, erkennt das System den Fehler sofort und ignoriert ihn. Gleiches gilt für alle anderen Befehlseingaben. Prinzipiell passieren nur korrekte Eingaben die Kommandozeile.

Die gewünschte Tonart teilen wir dem Extended Synthesizer System durch den entsprechenden Anfangsbuchstaben und gegebenenfalls das nötige Vorzeichen (Doppelkreuz beziehungsweise b) mit. Zum Beispiel c für C-Dur, c # für Cis-Dur, eb für ES-Dur und so weiter. Leider versteht die Software nur Dur-Tonarten. Moll-Tonarten müssen immer als Dur-Tonarten eingegeben werden. Dies läßt Vollblutmusikern natürlich die Haare zu Berge stehen.

Zu guter Letzt nun noch LOAD»DEMO«, 8 dann RETURN getippt und kurze Zeit später füllen sich unsere drei Notenzeilen. Ein Ausschnitt aus dem Musikstück erscheint notiert. Jetzt noch »PLAY« eingetippt und unser Extended Systhesizer System läßt sich nicht mehr halten. Flott trillert und flötet es dreistimmig vor sich hin. Noten flitzen über den Bildschirm. Auch optisch läßt sich mit dem Programm gewiß jeder Nachbar verblüffen. Nach zwei Minuten, Ruhe, der Spuk ist vorbei. Auf der Programmdiskette warten mehrere Demostücke auf Ihren Einsatz, alle recht passabel programmiert. Klanglich erlebt man jedoch hier weniger Überraschungen.

Wenig Sound, aber viel Musik

Im Vergleich zu Synthimat 64 oder Musicalc hat man bei der Konzeption des Extended Synthesizer Systems weniger Wert darauf gelegt, alle Soundmöglichkeiten des SIDs auszuschöpfen. Dafür erhält man hier mit Abstand das beste Werkzeug, um mit Noten zu hantieren. Leider lassen sich die erstellten Kompositionen nicht ausdrucken. Schade, mit dieser Option zusätzlich, hätte man ein passables Notendrucksystem für dreistimmige Kompositionen.

Ringmodulation, Synchronisation, Filterung und LFO-Modulationen muß man hier vergessen. Auch den DCOs erlaubt man hier nur Dreieck, Rechteck, Sägezahn oder Noise zu produzieren, keine der Mischformen.

Die Sounds legt man hier in sogenannten Registern ab. Insgesamt sind jeweils zehn verschiedene Register ohne Diskettenoperation abrufbar. Diese Register könnte man natürlich auch Klangpresets oder Klangprogramme nennen.

Man sollte bereits vor Beginn der Songprogrammierung über die Struktur des Songs möglichst im Klaren sein. Wer gut notieren kann, ist am besten dran. Er kann die Notation vom Papier direkt in den Computer »abschreiben«. Andernfalls dauert es sicher länger bis man ein ganzes Stück wohltönend einprogrammiert hat. Trial and Error lautet die Devise für alle nicht der Musiksprache fähigen.

Schnelles Komponieren dank sinnvoller Eingabeerleichterungen

Wie funktioniert die Noteneingabe? Zunächst bestimmt man die Stimme, die programmiert werden soll, mit den Befehlen 01, 02 und 03. Dann gibt man Ton für Ton jeweils Notenlänge, Tonhöhe, Lautstärke, Anschlag und das gewünschte Klang-Register ein. Der gerade eingegebene Ton erscheint sofort als Note in der richtigen Notenzeile.

Gott sei Dank braucht man nicht für jeden Ton erneut alle Werte eintippen, nur die, die sich gegenüber der zuvor eingegebenen Note ändern. Im anderen Fall übernimmt das System die entsprechenden Werte des vorangegangenen Tones. Ganz ungeduldige Klangignoranten dürfen getrost auch auf die Eingabe von Lautstärke, Anschlag und Register verzichten. Dann nimmt die Soft-



ware einfach gebräuchliche Ersatzwerte.

Eine vollständige Toneingabe sieht zum Beispiel so aus:

1.4 d5 : 9 : 1 : 3

Viertel Tonhöhe: Sustain: Anschlag Register Note: Oktave

Will man einen Wert nicht genau spezifizieren, gibt man statt dessen bei der ersten Noteneingabe ein »*« ein. Das System nimmt dann für diesen Parameter im gesamten Song einen Ersatzwert. Gibt man das Sternchen innerhalb eines Songs ein, übernimmt das System für diesen Parameter den Wert des vorangegangenen Tones.

Soll ein Ton in allen Parametern nochmals wiederholt werden, genügt ein Druck auf die RETURN-Taste. Dies erleichtert natürlich das Arbeiten ungemein.

Der Takt, der gerade eingegeben, bearbeitet oder gespielt wird, erscheint immer in seiner Umgebung, in allen drei Takten auf dem Bildschirm. Insgesamt kann das System 408 Takte pro Stimme im Speicher halten.

Die ersten acht Töne des bekannten »alle meine Entchen« erfordern einen doch recht hohen Programmieraufwand für eine Stimme.

Mit den Befehlen »+« und »-« kann man die Takte horizontal über den Bildschirm scrollen. Einige weitere nützliche Befehle dienen dazu, etwa einen schon eingegebenen Takt a gezielt auf den Bildschirm zu rufen (list a), bis zum Takt e zu löschen (del a-e) oder i-mal zu kopieren (copy a-e,i). Takt-Handling nennt man das. Auch die Funktionstasten f1, f3, f5 und f7 bieten interessante Arbeitserleichterungen. Ein einziger Tastendruck löscht den zuletzt eingegebenen Ton (f1) oder Takt (f3), kopiert den letzten Takt (f5), oder setzt einen ganzen Takt beziehungsweise den Rest des gerade Eingegebenen als Pause (f7). So geht die Eingabe der Kompositionen trotz komplexer Befehle relativ schnell vonstatten.

Mehrmaliges Wechseln von Tonart und Tempo innerhalb einer Komposition stellt kein Problem dar. Hierfür sind die Befehle »key« und »tempo«, mit nachfolgender Eingabe des Tonartsymbols beziehungsweise des Tempowertes zuständig. Sowohl Tempo- als auch Tonartänderungen beziehen sich nicht auf eine komplette Stimme sondern auf eine wählbare Anzahl von Takten. So sind mehrere Änderungen innerhalb eines Stückes möglich.

Die Einstellungen der einzelnen Klangregister gibt man numerisch ein. Wir können die Wellenformen und die ADR-Werte für jedes Register getrennt bestimmen. Die Wellenform legt man mit »t« für Dreieck, »p« für Rechteck, »s« für Sägezahn und »n« für Noise, fest. Als nächstes bestimmt man die ADR-Werte durch Eingabe des Buchstabens A, D beziehungsweise R und einer darauffolgenden Zahl von 0 für kurz, bis 9 für lang. Der Sustainwert steht nicht im Klangregister. Ihn gibt man bei der Toneingabe für jeden Ton getrennt ein. Was musikalisch natürlich gutzuheißen ist. Denn so lassen sich einzelne Töne mit unterschiedlicher Lautstärke spielen, dynamisch. Dies klingt weniger mechanisch als wenn alle Töne gleiche Lautstärke besitzen.

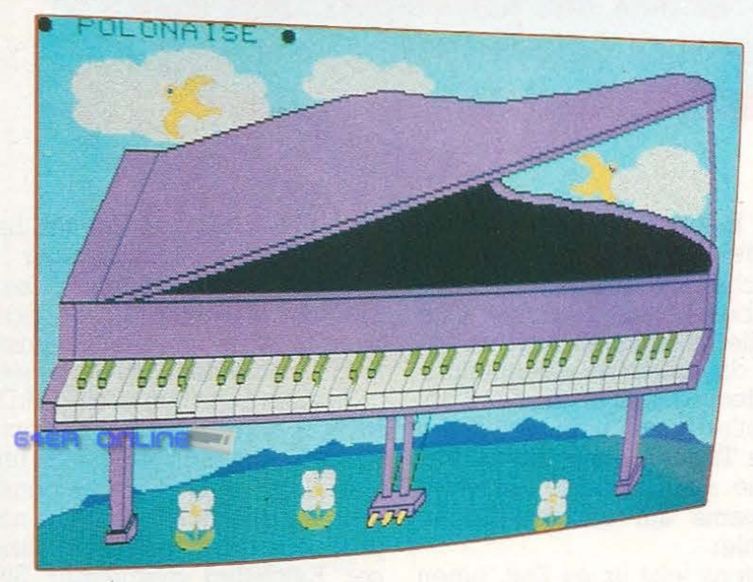
Das System läßt sich übrigens auch stimmen.

Obwohl der Extended Synthesizer weder die Ringmodulations- noch Synchronisationsmöglichkeiten des SIDs nutzt, erreicht man doch ganz interessante Soundabläufe. Jeder einzelne Ton läßt sich ja mit einem eigenen Sound versehen. Das gleicht das Fehlen besagter Effekte mehr als aus.

Die Lautstärkenwerte, Gate-Zeiten und angewählten Register der einzelnen Töne zeigt die Software, sofern gewünscht, auf dem Bildschirm.

Solange man mit dem System noch nicht so ganz hundertprozentig vertraut ist, passieren natürlich des öfteren Eingabefehler. In diesem Falle läßt uns das System nicht ratlos alleine, sondern hilft mit diversen Fehlermeldungen meist aus der Patsche. Dies gilt natürlich nur für falsch, beziehungsweise unvollständig eingegebene Kommandos. Klingt unser Lied am Schluß fürchterlich, so müssen wir das entweder mit unserem musikalischen Gewissen verantworten, oder tiefer in die Grundlagen der Harmonielehre einsteigen. Ein solches musikalisches Gewissen besitzt das Extended Synthesizer System nicht.

Will man sich seinen Song anhören, tippt man nur den Befehl PLAY ein und schon geht's los. Auch einzelne Takte, sowie einzelne Stimmen spielt der Extended Synthesizer auf Wunsch vor. Sogar an eine Loop-Funktion, die einzelne Takte sowie eine oder mehrere Stimmen zyklisch wiederholt, hat man gedacht.



Übrigens: Dieses Piano wurde nicht mit dem Extended Synthesizer erzeugt.

Für die Nachwelt können wir unsere Songs sowohl auf Diskette als auch auf Cassette aufbewahren. Zu jedem Song werden zugleich sämtliche Klangparameter mit abgespeichert.

Ein Aspekt, Extended Synthesizer einzusetzen, ist das Üben mehrstimmiger Notation. Wenn man nach dem Eintippen den Song abspielt, hört man sofort, ob man seine musikalische Imagination richtig zu »Papier« gebracht hat oder nicht.

Das Extended Synthesizer System ist vor allem für Notationsfetischisten und Composerfreaks gedacht, denen Effekte wie Ringmodulation und Synchronisation sowie viele LFOs und Realtimeeinspielung, nicht so wichtig sind. Man muß hier jedoch auch bemerken, daß die Notendarstellung nicht bis ins kleinste Detail der Hohen Schule der Notation entspricht. Doch an diesem Problem beißen sich momentan noch ungleich leistungsfähigere Computersysteme die Zähne aus. Das Programm kostet 138 Mark.

(Richard Aicher/aa)

Übersicht der Musikprogramme für den C 64

	MUSI-CALC	EXTEN-DED SYN-THESIZER SYSTEM	SYNTHI-MAT	ULTI-SYNTH	SEQUEN-ZER 64	SYNTHY 64	MUSIC MACHINE	MULTI-SOUND SYNTHE-SIZER 64
VCO	3	3	3	3	3	3	3	3
Kurvenformen	D/S/R	D/S/R	D/S/R + 4 Misch-formen	+	—	D/S/R	D/S/R	D/S/R
LFO	—	—	8	—	2	—	—	1 (VCO3)
Noise	+	+	+	+	+	+	+	—
ADSR	3	3	3	3	3	3	3	3
Pulsweite	+	—	+	+	+	+	—	3
Typ	L/B/H	—	L/B/H + 4 Misch-typen	L/B/H	L/B/H	L/B/H + 4 Misch-typen	—	L/B/H + Notch-filter
Filter Freq	+	—	+	—	+	+	?	+
Res	+	—	+	—	—	+	?	+
Ring-Modulator	+	—	+	+	+	+	—	+
Sync	+	—	+	+	+	+	—	+
Synchronisat extern	+	—	—	—	Trigger in	—	—	—
Stimmen spielbar auf Keyboard (Live)	1	—	[illegible]	[illegible]	1	—	?	1
Einspiel-Sequenzer	je Durch-gang 1 Stimme maximal 3	—	3 Stimmen gleichzeitig maximal 3	Je Durch-gang 1 Stimme maximal 3	Je Durch-gang 1 Stimme	—	Je Durch-gang 1 Simme, maximal 2	Je Durch-gang 1 Stimme, maximal 2
Composer System	+	+	—	+	+	+	+	—
Songbildung	+	—	—	+	+	+ (mittels Programm-schleifen)	—	—
Portamento	—	—	—	+	+	+	—	—
Notation:								
Schirm	+	+	—	—	+	—	+	—
Ausdruck	+	—	—	—	+	—	—	—
Presets momentan verfügbar:								
Sounds	32	10	256	k.A.	16	beliebig	k.A.	k.A.
Songs	32 (je 280 Töne)	1	1	1	48 Takte	1	1	1
Preis	500.- (3 Disket-ten)	138.-	99.-	79.-	100.-	k.A.	59.-	k.A.
Datenträger		Disk	Disk	Disk/Cass	Disk	Cass	Cart	Disk
Sonstiges	*1)	*2)	*3)	*4)	*5)	*6)	*7)	*8)

Sonstiges:
*1) 70 Keyboard-Layouts/Externe Programme, jede Taste kann eigens gestimmt werden, Synchronisation mit externen Rhythmusgeräten vorgesehen.
*2) Drei Stimmen werden in Realtime gleichzeitig am Bildschirm notiert.
*3) Songs werden direkt auf Diskette aufgenommen. Für jeden VCO stehen zwei seperate LFOs zur Verfügung. Je ein LFO kann den VCA beziehungsweise VCF modulieren.
*4) Modulationsoszillator mit beliebigen Kurvenformen belegbar. Eine Stimme Realtime spielbar, zwei Stimmen als Playback zu programmieren.
*5) Triolen, Quintolen, Septolen spielbar, Notendruck mit Zusatzprogramm
*6) Parallel zur Musikausgabe kann Bildschirmgrafik programmiert werden, Toneingabe in einer Art Musikbasic.
*7) 7 Rhythmuspresets zur Begleitung. Keine Notenschlüssel. Keine erniedrigten Töne möglich.
*8) Maximal zwei Stimmen gleichzeitig abspielbar, 8 presets für 1 Background-Rhythmus- oder -Melodiestimme. VCO3 und ADSR 3 modulieren VCO1,2 sowie VCF-Parameter.

Quickcopy – das schnelle Kopierprogramm für den C 64

Schon lange war bekannt, daß es Möglichkeiten gibt, die Floppy VC 1541 schneller zu machen. Beim gelegentlichen Laden und Speichern von Programmen störte die bekannte Trägheit des Laufwerks weniger. Wenn allerdings viel kopiert werden sollte, wurde sie zur Geduldsprobe. Jetzt gibt es ein schnelles Kopierprogramm, das zu einem erschwinglichen Preis erhältlich ist.

Schon seit einiger Zeit tauchten in allen möglichen Computerzeitschriften Anzeigen auf, die ein schnelles Kopieren von Programmen mit dem C 64 versprachen. Genauer gesagt waren die dort angegebenen Kopierzeiten sensationell. Selbst die unter der Hand herumgereichten schnellen Kopierprogramme von »Spezialisten« konnten nicht mit dieser Geschwindigkeit aufwarten. Als wir endlich eine Testversion bekamen, wurde alles liegengelassen um festzustellen, wo der Haken lag.

Um es vorwegzunehmen und ohne Schöntuerei: Es gibt keinen Haken. Quickcopy ist ein sehr schnelles und ausgereiftes Kopierprogramm (für den C 64, versteht sich).

Geliefert wird Quickcopy auf einer Diskette mit einer ausführlichen Beschreibung, die an sich gar nicht notwendig ist. Kopierprogramme sind ja normalerweise unkompliziert zu bedienen, vor allem diejenigen, die nicht einzelne Files kopieren, sondern komplette Disketten, sogenannte Backup-Versionen erstellen. Deren einzige Aufgabe ist es, ein Duplikat einer Diskette herzustellen. So lassen sich auch mit Quickcopy keine einzelne Files kopieren, sondern nur ganze Disketten. Trotzdem können einige Parameter eingestellt werden: Da Quickcopy wahlweise mit einem oder auch zwei Laufwerken arbeitet, kann deren Geräteadresse bestimmt werden. Anschließend wird der Kopiermodus festgelegt. Man kann wählen zwischen normalem und Utility- Modus. Im normalen Modus werden nur die als belegt gekennzeichneten Sektoren, im Utility-Modus alle Sektoren kopiert. Letzteres ist immer dann sinnvoll, wenn zum Beispiel die Block Availability Map (BAM) fehlerhaft ist oder um auch gelöschte Files mitzukopieren. Da im Normal-Modus nur die belegten Blöcke kopiert werden, ist eine Kopie je nach Belegungsgrad der Diskette langsamer oder schneller hergestellt. Für jeden Parameter werden Standardwerte vorgegeben, auch ob ein Verify gewünscht wird oder ob Lesefehler ignoriert werden sollen (wichtig bei beschädigten Disketten), so daß durch sechsmaliges Drücken der Returntaste sämtliche Standardwerte eingestellt werden und der Kopiervorgang anläuft. Falsch machen kann man eigentlich gar nichts. Fehler, etwa der Schreibschutz auf der Diskette werden abgefangen und lassen das Programm nicht abstürzen.

Bei einigen Kopierprogrammen wird während des Kopierens der Bildschirm abgestellt (nicht gelöscht), um schneller zu werden. Bei Quickcopy hingegen ist während des gesamten Kopiervorgangs der Bildschirm sichtbar. Interessant ist, daß angezeigt wird, welcher Track und Sektor gerade in Bearbeitung ist. Außerdem wird die Art der momentanen Tätigkeit angezeigt, ob Formatieren, Lesen, Schreiben oder Verifizieren. Interessant dabei ist die Reihenfolge der Bearbeitung: Jeder Block wird zuerst formatiert, dann beschrieben und zum Schluß verifiziert, anders als bei herkömmlichen Kopierprogrammen, bei denen zuerst die gesamte Diskette formatiert und erst danach mit dem Kopieren begonnen wird. Und alles geht mit einer bisher nicht gewohnten Geschwindigkeit vonstatten. Zwei extreme Beispiele:

1. Die Quelldiskette ist formatiert aber leer: Kopierzeit = 27 Sekunden, einschließlich Formatieren, Verifizieren und Diskettenwechsel.

2. Die Quelldiskette ist komplett voll: Kopierzeit 4,5 Minuten inklusive Formatieren, Verifizieren und dreimaligem Diskettenwechsel.

Ein Verzicht auf Formatierung lohnt sich nicht, der Vorteil liegt im Sekundenbereich. Der Verzicht auf ein Verify bringt ungefähr eine halbe Minute und darauf sollte aus Sicherheitsgründen auch nicht verzichtet werden.

Alles in allem ist Quickcopy ein Programm, das längst fällig war. Und auch die Tatsache, daß mit ihm die meisten kopiergeschützten Programme nicht dupliziert werden können (ich habe es ausprobiert), ist keine wesentliche Einschränkung, im Gegenteil, es zeugt von Verantwortungsbewußtsein.

Dieses Programm zeigt und läßt erwarten, daß es bald möglich sein wird, die Floppy VC 1541 insgesamt schneller zu machen. Und darauf kann man gespannt sein. (gk)



— oder was wirklich im Commodore 64 steckt

Musicalc ist eine elegante Methode, mit einem Commodore 64, Töne, Klänge, Kompositionen zu speichern und wiederzugeben. Noten erscheinen am Bildschirm — mit weiteren C 64 oder einem Elektronikschlagzeug bleiben wir stets im Takt. Wem die heimische Musik zu langweilig wird, der holt sich eine der 50 exotischen Tonleitern in den Speicher und spielt zum Beispiel Bali Agung.

Die Disketten von Musicalc stecken nicht in den altbekannten, tristen schwarzen Hüllen, sondern präsentieren sich farbenprächtig, jede ein kleines grafisches »Kunstwerk« für sich. Jede ist über dies in einem ausgesprochenen hübsch gestalteten Umschlagskarton verpackt. Hier waren Optikspezialisten am Werk.

Zum Test lag mir leider nur der erste von drei Bänden, in der amerikanischen Version, vor. Band ist das passende Wort. Auf 65 Seiten großformatigem Druck erfährt man alles Wichtige über die Diskette Nummer 1, den Musicalc Sequenzer und Synthesizer. Grundlegende Themen der Synthesizer kommen zur Sprache. So können sich auch in diesem Gebiet noch völlig Unbedarfte ohne Probleme ins Reich des SID begeben. Für Fortgeschrittene oder Ungeduldige beinhaltet das Handbuch eine »Schnell-Anleitung« mit den wichtigsten Befehlen in Kurzform. Musicalc ist kein abgeschlossenes Programm, sondern ein ganzes Programmsystem, das ständig um weitere Soft- beziehungsweise Hardware erweitert wird. Dies verspricht zumindest der Hersteller, die amerikanische Firma Waveform. Sämtliche, auch in Zukunft erscheinende Softwaremodule, bleiben untereinander kompatibel. Problemlos lassen sich Daten innerhalb des Systems austauschen. Neben neuen Programmen erscheinen in Zukunft auch ständig neue Demostücke auf Disketten. So bringen auch Programmier- oder Spielfaule ihren Commodore 64 zum Tönen. Zwei dieser sogenannten Templates gibt es schon. Das African/Latin Rhythm Template und das New Wave and Rock Template.

Ich möchte die ersten drei Programmodule des Musicalc-Systems (typischer Hardwareaufbau siehe Bild 1) besprechen, drei Disketten, randvoll mit Software, die Sequenzer und Synthesizer-Diskette Nummer 1, die Score-Writer-Disk Nummer 2 und schließlich Keyboardmaker, das Programmodul 3.

Die Sequenzer- und Synthesizerdiskette stellt den Grundbaustein im Musicalc-System dar, den Schlüssel zur Welt des Klangs und der Melodie.

Bei Musicalc symbolisiert die linke Bildschirmhälfte (Bild 2) das Einstellfeld eines ganz gewöhnlichen Synthesizers mit Schiebereglern und verschiedenen Schaltern, das Karofeld rechts einen Multisequenzer, der drei Stimmen gleichzeitig anzeigt. Bei beiden Einstellfeldern hielten sich die Softwareentwickler nahe an Vorbilder der klassischen Analogsynthesizertechnik. Dies macht es auch Musikern einigermaßen leicht, Musicalc Töne zu entlocken. Der vertraute Anblick eines Synthesizers und Analogsequenzers bleibt wenigstens in etwa am Bildschirm erhalten.

Unbedingt erforderlich ist ein Farbbildschirm. Verschiedene Hintergrundfarben signalisieren nämlich die diversen Bedienungsebenen, in denen die Commodore-Tastatur jeweils unterschiedliche Belegung aufweist.

Ordnung in die vielen verschiedenen Modi von Musicalc bringt das Hauptmenü. Ich möchte nur auf einige wichtige Optionen eingehen. Das genügt, um einen Eindruck von den vielen Möglichkeiten zu verschaffen.

32 verschiedene Soundeinstellungen und 32 verschiedene dreistimmige Melodielines können mittels »SAVE PRESETS« auf ein gemeinsames Preset-File abgelegt werden.

Erarbeitete Melodien und Klangeinstellungen können mittels »SAVE PRESETS« beziehungsweise »LOAD PRESETS« auf Diskette gespeichert oder in den Computer geladen werden. Auf jedem File haben 32 Sequenzen und 32 Sounds Platz. Zwei Demo-Preset-Files erhalten wir bereits gratis auf der Programmdiskette 1 mitgeliefert. Waveform hat sich sowohl mit der Auswahl der Demostücke als auch den Klangeinstellungen Mühe gegeben. Selbst ganz moderne Klänge tauchen hier und da auf. Andere Sound- und Sequenzereinstellungen erinnerten mich wiederum sehr an Klänge der altbekannten Elektronik-Rocker »Tangerine Dream«.

Für unsere ersten Versuche laden



wir ein Demofile in den Arbeitsspeicher. Dann begeben wir uns mittels der Option »PRESETS« in den Preset-Spiel-Mode. Auf dem Bildschirm unser Bild 2. Grüner Hintergrund signalisiert: Presetmode! Jetzt wartet der Commodore auf unseren Einsatzbefehl. 32 verschiedene dreistimmige Sequenzen und 32 unterschiedliche Klangeinstellungen stehen abrufbar im Arbeitsspeicher bereit. Jeder von 32 QWERTY-Tasten ist jeweils eine Melodie und ein Sound zugeordnet. In der Melodieebene (Commodore Taste) rufen wir durch Druck auf einer Taste entweder die zugehörige Melodie, in der Soundebene (Shift-Taste) den zugehörigen Sound auf. Die Änderung erfolgt sofort, ohne Pause im Spiel. Jede der 32 Sequenzen läßt sich mit jedem der 32 Klänge versehen. Ohne Diskettenoperation stehen also momentan 1024 Kombinationen (32*32) von Sounds und Sequenzen abrufbereit zur Verfügung!

Demomelodien und -klänge abspielen ist bequem, aber auf die Dauer natürlich nicht beglückend. Irgendwann wächst der Wunsch, eigene Klänge und Melodien zu kreieren. Und jetzt kann man so richtig loslegen.



▲ Bild 1. Commodore 64 und ein Analogsynthesizer spielen im Duett — erstmals möglich mit Musicalc

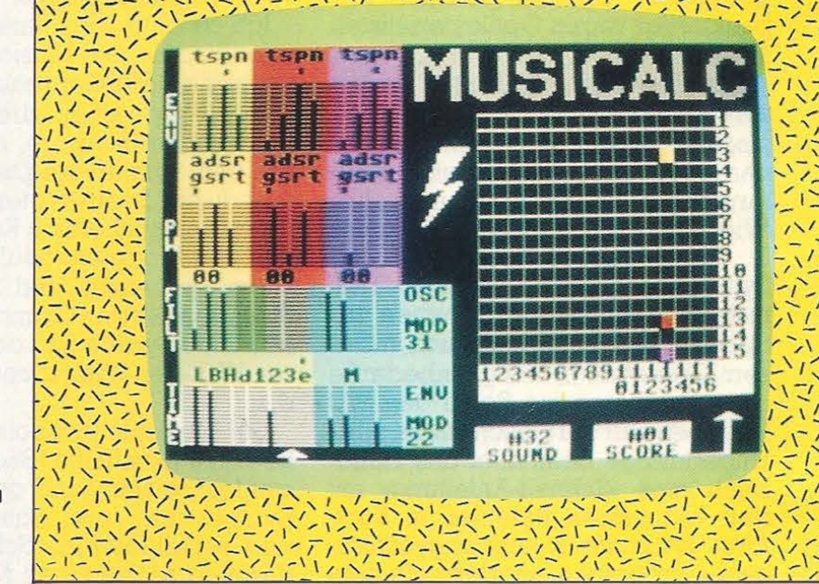


Bild 2. Das Preset-Panel von Musicalc — man denkt, man hat einen Synthesizer vor sich ►

70 Schalter und Regler stehen zur Verfügung

Im Syntheziser-control-panel, mit grauer Hintergrundfarbe, stellen wir die Klänge ein. Die Klangeinstellung funktioniert prinzipiell genau wie bei einem Analogsyntheziser mit Schiebereglern. Insgesamt regeln wir 70 verschiedene Klang-Parameter über das Panel. Es existierten sowohl »Schieberegler« als auch »Schalter«. Die Schieberegler erscheinen als vertikale schwarze Balken, die Schalter als kleine Vierecke im Bild. Mit unseren QWERTY-Tasten wählen wir die einzelnen Schieber und Schalter an. Entsprechend der Sound- und Melodiewahl rufen wir wieder jeweils einen Schieber und Schalter über eine Taste auf.

Mit den »Schiebereglern« stellen wir im Bereich ENV (linke obere Ecke im Bildschirm), die Hüllkurven ein. Hierzu wählen wir für jeden der drei Generatoren den entsprechenden Attack-, Decay-, Sustain- und Releasewert. Jede Stimme können wir mit einer anderen Hüllkurve versehen. Unter dem ENV-Bereich das PW-Einstellfeld. PW steht für Pulsweite. Wieder einen Stock tiefer, mit FILT gekennzeichnet, das Filter-Einstell-Feld. Jede der drei Stimmen verfügt über einen getrennt regelbaren Filtereinsatzpunkt, die Resonanz des Filters ist ebenfalls regelbar. Mit der untersten Reglerreihe können wir die Geschwindigkeiten der drei Sequenzerkanäle regeln.

Neben den »Schiebereglern« existieren noch zirka 30 Schaltfunktionen. So lassen sich für jeden der drei Oszillatoren getrennt verschiedene Kurvenformen auswählen: Dreieck, Sägezahn, Rechteck beziehungsweise Rauschen. In der obersten Zeile unseres Panels in Bild 2 lesen wir dreimal »TSPN«. Dies sind nichts anderes als die ersten zwölf Schalter, mit denen also Triangle (Dreieck), Sawtooth (Sägezahn), Pulse (Rechteck) und Noise (Rauschen) für jeden der drei Oszillatoren an- und ausgeschaltet werden. Im Einschaltzustand erscheint jeweils ein schwarzes Viereck unter dem entsprechenden Symbol. Mit einer weiteren Schaltserie (3 mal GSRT!) lassen sich Gate, Synchronisation und Ringmodulation ein- beziehungsweise ausschalten. Die letzte Schalterreihe ermöglicht die Auswahl der gewünschten Filtereinstellung. Insgesamt stehen acht verschiedene Filtereinstellungen zur Verfügung: Tiefpass, Bandpass, Hochpass und die bekannten Mischformen.

Das Motto: Probieren geht über

studieren, gilt für die schier unzähligen Möglichkeiten der Modulation. Jeder Oszillator und Envelopegenerator kann so ziemlich jeden anderen Parameter modulieren. Dies eröffnet eine uferlose Anzahl von Klangvariationen. Die einzelnen Modulationsarten lassen sich durchschalten und sofort in ihrer Auswirkung auf den Sound überprüfen.

Hat man nach Stunden voller Mühe den richtigen Sound gefunden, geht's ans Ausarbeiten einer Sequenz, Melodie, beziehungsweise Komposition. In der rechten Bildschirmhälfte unseres Hauptpanels (Bild 2): Das Sequenzersystem. Es besteht aus 15 waagrechten Reihen mit jeweils 16 Ton-Schritten. Jeden der 15 x 16 Kästchen entspricht ein bestimmter, frei wählbarer Ton. 240 Töne faßt jeder der 32 Sequenzer-Presets maximal. Drei farbige Quadrate entsprechen den drei Sequenzerstimmen. Sie durchlaufen nun Reihe für Reihe, Kästchen für Kästchen, das gesamte Feld oder auch nur Teile davon. Bei jedem Schritt erklingen maximal drei Töne im gewählten Sound. Keine Einschränkungen gibt es für die Art des Durchlaufs. Wir können die Kästchen entweder brav Reihe für Reihe laufen lassen. Aber auch wirre Sprünge von jeder beliebigen Stelle zu jeder anderen sind möglich. So können wir zum Beispiel Stimme 1 endlos Reihe 10 wiederholen lassen, während Stimme 2 die Reihen 1 bis 16 zyklisch durchläuft und Stimme 3 Reihe 3 von step 4 bis 8, dann Reihe 4 von step 1 bis 8 und schließlich Reihe 10 von step 8 bis step 12 spielt. Nichts ist unmöglich. Ein fast perfektes Sequenzersystem.

Welche Töne den 256 Kästchen entsprechen sollen, bestimmen wir in der Option SCORE. Hier entsteht quasi unsere Partitur. Das Score-Eingabefeld sehen wir in Bild 3.

Mit einem Cursor lassen sich hier in einer Art Balkendiagramm die Tonhöhen und Oktavlagen der jeweils 16 Töne einer Sequenzerzeile einstellen. Insgesamt existieren für jedes der 32 Sequenzerfelder eines Preset-Files 16 solche Scores. Das Komponieren von Songs sollten Ungeduldige lieber bleiben lassen, es erfordert Zeit.

Wem diese Art der Melodieentwicklung zu langsam vorangeht, der kann jedoch seine Sequenzen auch im Realtime-Verfahren, über Commodore 64-Tastatur, einspielen.

Hierzu wählt man im Hauptmenü die Option KEYBOARD. Drei weitere Unteroptionen stehen jetzt zur Wahl.

Im RECORDER MODE kann jeweils eine der drei Sequenzerstimmen per Hand eingespielt werden, die zwei restlichen hört man hierzu als Begleitung. Jeder Tastendruck wird registriert und überschreibt die ausgewählte Sequenzerspur an der bestimmten Stelle. Berührt man keine Taste, übernimmt der Sequenzer die ursprünglichen Töne.

Im STEP MODE lassen sich bestimmte, einzelne Steps innerhalb der Sequenz gezielt verändern, und zwar durch Eingabe des gewünschten Tones über das Keyboard.

Im sogenannten VOICE MODE können wir eine der drei Stimmen des Sequenzers abschalten und per Hand über das C 64-Keyboard spielen. So spielen wir dann ein Solo zu unserem zweistimmigen, vorher programmierten Begleitorchester.

Noch viele weitere Modi existieren. Auf alle hier einzugehen, würde den Rahmen doch erheblich sprengen. Eines möchte ich noch erwähnen. Im Mode EXTERNAL können einige externe Programme geladen werden. Mit einem hiervon lassen sich zum Beispiel die Oszillatoren des Commodore 64 in weitem Bereich stimmen. Der jeweilige Frequenzbereich erscheint numerisch, auf vier Stellen genau, im Bildschirm. Musicalc als Stimmgerät.

Die Diskette 2 nennt sich Score Maker. Sie dient zur Übersetzung der im Sequenzer befindlichen Songs in Notenschreibweise. So erscheinen Noten am Bildschirm. Mit einem VC 1525 Grafikdrucker oder über einen Epson mit Cardco Centronics-Interface lassen sich diese ausdrucken. Bei meiner Testversion gab es hier jedoch einige Schwierigkeiten. Ein perfekter Ausdruck gelang nicht.

Mir lag für diesen Teil von Musicalc allerdings noch keine schriftliche Bedienungsanleitung vor. Da sich das gesamte Programm mittels ausführlicher Help-Texte (Bild 4) von selbst erklärt, ist diese normalerweise auch nicht nötig. Vielleicht

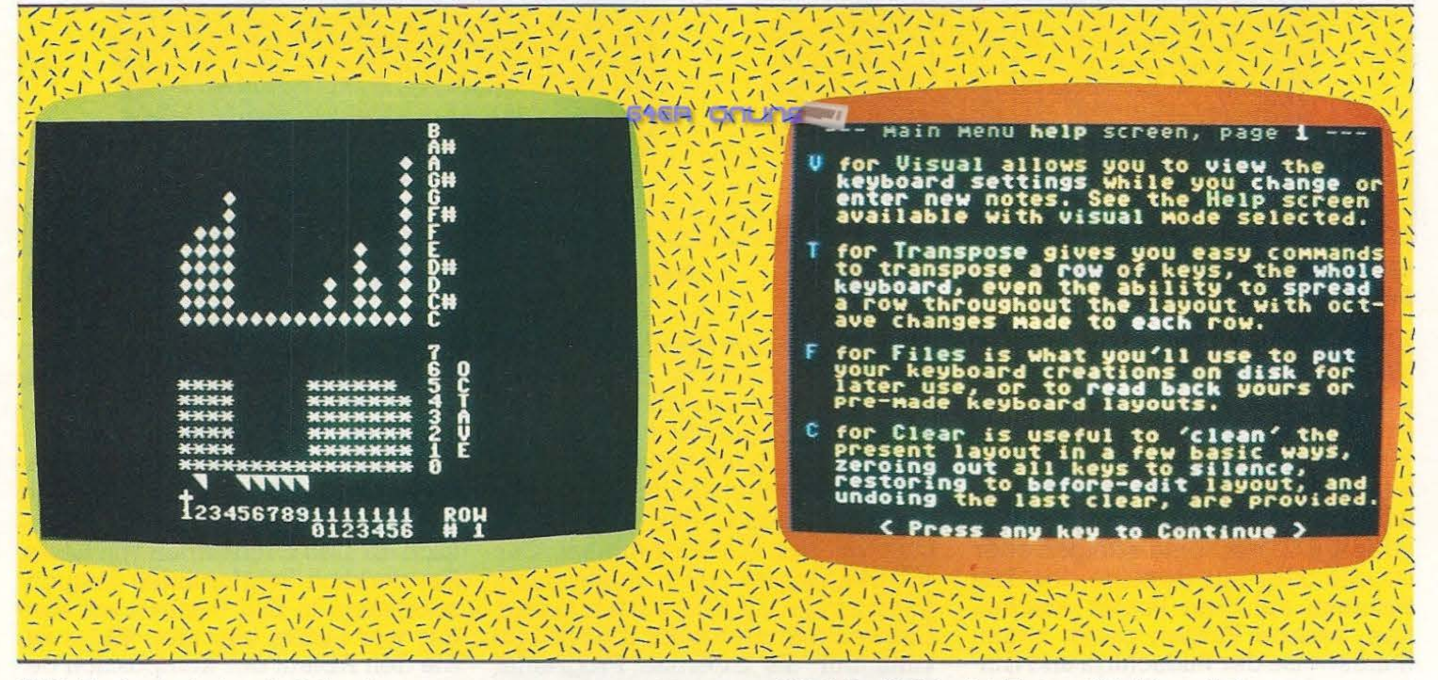


Bild 3. Toneingabe in einer Art Balkendiagramm am Bildschirm

Bild 4. Ausführliche Help-Texte, natürlich in englisch, erklären die Bedienbefehle



fehlen jedoch hier genauere Hinweise auf diverse Druckkommandos.

Sämtliche Programme auf dieser Diskette funktionieren natürlich nur in Verbindung mit der Diskette 1. Denn nur ausschließlich mit der Synthesizer- und Sequenzerdiskette erstellte Kompositionen versteht der Scorer.

Nach dem Start liest man von der Diskette 1 oder einer anderen Diskette mit erstellten Presetfiles ein gewünschtes Preset ein, gibt an, welchen der 32 Songs man notieren will, wartet, bis der Song geladen ist und legt dann wieder die Score-

farben. »Ich bin vollauf beschäftigt«, signalisiert die Software. Die Berechnung einer Doppelzeile dauert zirka eine Minute. Dann erscheint Bild 5 am Schirm.

Kurze Zeit später sollte der Drucker die Zeilen ausdrucken. Wie gesagt, tat er dies, trotz mehrmaligen Versuchen mit diversen Druckern, nicht.

Auf der Score-Writer-Diskette befinden sich noch weitere sehr wichtige Programme.

Das erste ist ein Synchronisationsprogramm, das den Commodore 64 mit weiteren C 64 oder auch einer Rhythmusmaschine synchronisiert.

Das zweite wichtige externe Programm nennt sich LIST MAKER. Es erweitert den Sequenzer der Musicalc-1-Diskette, macht längere und komplexere Sequenzen möglich. LIST MAKER kann man als »Sequenz-Sequenzer« bezeichnen. Er bildet ganze Kompositionen aus den 32 Sequenzen und Sounds eines Presets. 64mal lassen sich Klang und Melodie hintereinander wechseln, wobei man noch jeweils bestimmen kann, wie viele 16er-Zyklen einer Sequenz-Sound-Kombination er hintereinander durchläuft, ehe LIST MAKER zur nächsten Kombination wechselt. So bilden wir durch

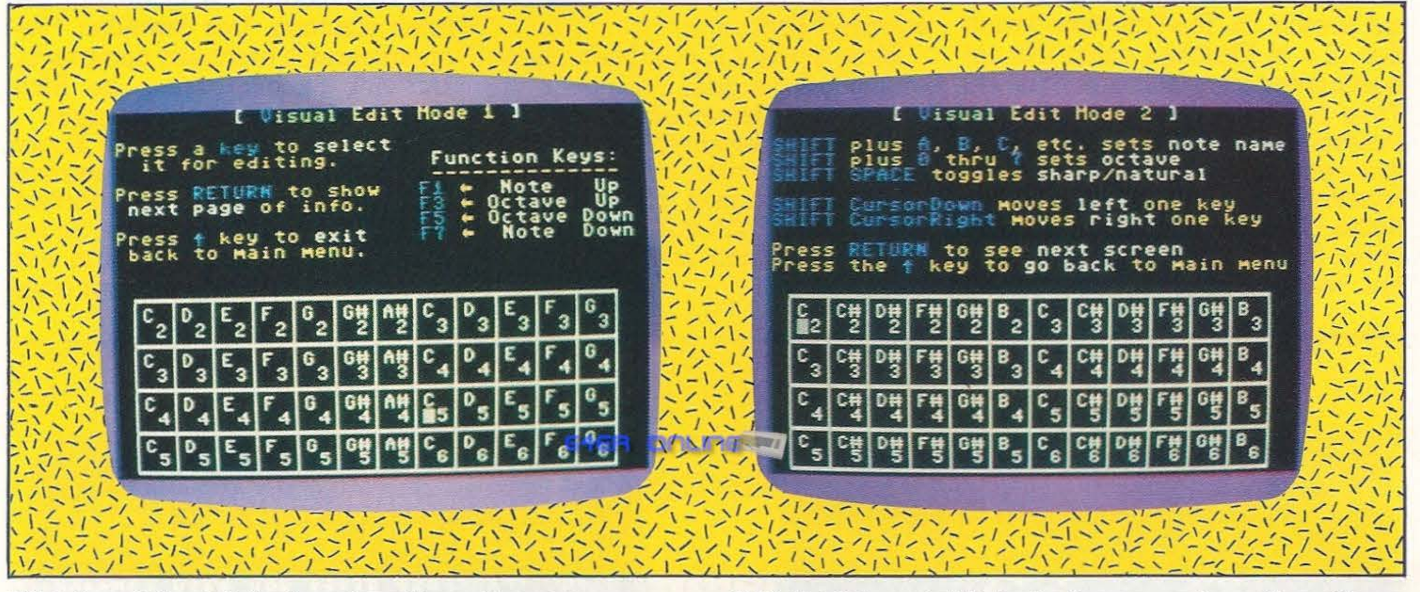


Bild 5. Score Writer stellt die eingegebenen Kompositionen am Bildschirm Zeile für Zeile in Notenschrift dar

Bild 6. List Maker verknüpft einzelne Sequenzen zu langen Kompositionen

Writer-Diskette ein. Das Score-Writer-Programm wird nun wieder geladen. Das Arbeiten mit diesen Programmen verlangt leider viele Wartepausen und öfteres Diskettenwechseln. Als nächstes bestimmen wir, ob der Ausdruck im 2/4, 3/4 oder 4/4 Taktschema erfolgen oder ob alle oder nur einzelne der drei Stimmen auf dem Papier erscheinen, welche Farben diese, beziehungsweise der Bildschirm und Hintergrund besitzen sollen.

Dann geht's ab in den Printmode. Zwei verschiedene Druckmodi existieren. Einmal der sogenannte AUTO PRINT-Mode, in dem die gesamte Sequenz Zeile für Zeile bis zum Schluß ausgedruckt wird. Daneben gibt es den MANUAL PRINT-Mode. Hier können einzelne Sequenzteile ausgedruckt werden. Der Druckvorgang stoppt nach jeder fertigen Doppelzeile.

Hat man den Printvorgang angewählt, beginnt auf dem Bildschirm ein flackerndes Farbenspiel aus schnell wechselnden Hintergrund-

Dies ist bisher ein Novum im Bereich der Musiksoftware für den Commodore 64. So läßt er sich nun endlich in weiteres Equipment integrieren. Denn ohne Synchronisation spielt das Schlagzeug schon längst auf »und«, während die 64-Sequenz noch mit der »Eins« zu tun hat. Das »C 64-Quartett« wird sicher nicht mehr lange auf sich warten lassen. Dieses Programm heißt E. SYNC. Es kann nur als externes Programm über die Musicalc-1-Diskette aufgerufen werden. Man kann dann entweder den C 64 mit einem extern eingegebenen, zum Beispiel von einem Elektronik-Schlagzeug stammenden, Sync-Impuls synchronisieren, oder ein TTL-Clock-Signal vom C 64 an eine externe Einheit abgeben. Die Signale werden über den User-Port eingeschleust, beziehungsweise dem Expansion-Port ausgegeben. Doch Vorsicht beim Hantieren an den Ports! Bei unsachgemäßem Vorgehen können Baugruppen des Computers leicht ihren Geist aufgeben.

Aneinanderhängen von einzelnen Sequenzen komplexe Musikstücke mit vielen Sound- und Melodieabschnitten. Die fertigen Kompositions-Listen speichern wir, mit Namen versehen, auf Diskette und spielen sie nach Belieben automatisch ab. Das Orchestrion von heute. Ein wirklich komfortables Sequenzersystem. In Bild 6 sehen wir eine der vier Listen, hier Step 49 bis 64, die den Ablauf der Komposition bestimmen.

Keyboardmaker (Bild 7) nennt sich die dritte bisher im Musicalc-System erschienene Programmdiskette. Dies ist das Programm für alle Liebhaber der Musik fremder Kulturkreise. Hiermit spart man sich das Auswendiglernen und Einüben von Tonleitern. Diese Aufgabe übernimmt ab sofort der Commodore 64.

Die erste Option des Hauptmenüs heißt VISUAL.

Im oberen Bildschirmteil befindet sich ein Bedienungsmenü, darunter die QWERTY-Tastatur des Commodore 64, versehen mit der aktuellen

Tonbelegung. Drückt man eine Taste, erscheint auf dem Bildschirm im zugehörigen Feld ein blinkender Cursor. Der entsprechende Ton klingt aus dem Lautsprecher. Soweit nichts Neues — das gab es schon oft. Die Tastenbelegung ist jedoch hier nicht fix, sondern beliebig wählbar. Im Bereich von sieben Oktaven können wir jeder Taste einen beliebigen Ton zuweisen. Die gewünschte Oktavlage gibt man mittels SHIFT plus 1,2,3... ein, die Töne mittels SHIFT plus C, D, E ... Der Ton »Des« läßt sich nur durch Erhöhung eines »C« darstellen. Es existiert kein Erniedrigungszeichen. Ein Manko in diesem Programmteil: Im Live-Spielmodus arbeitet es zu langsam. Spielt man zwei aufeinanderfolgende Töne etwas schneller, hinkt die Tonerzeugung meist etwas nach. Deshalb ist dieser Programmteil zum Spielen von schneller Melodik unbrauchbar. Die Keyboard-Layout-Daten lassen sich jedoch in das Sequenz-Programm der Diskette 1 transferieren. Spielt man hier dann im VOICE-Modus eine Melodie auf der QWERTY-Tastatur, klingen die einzelnen Töne entsprechend dem neuen Tastaturlayout und ohne Verzögerung.

Alle Keyboardlayouts lassen sich natürlich auf Disketten ablegen.

Ein hervorragender Einfall der Waveform-Leute war, zirka 50 Tonleitern aus allen möglichen Kulturbereichen als fertige Keyboardlayouts mitzuliefern. Dies stellt wirklich eine wahre Fundgrube für alle Liebhaber der Musik ferner Völker dar. Einen Teil der mitgelieferten Tonleitern sehen wir in Bild 8 und 9.

Leider wurde nicht berücksichtigt, daß nicht alle Völker mit unserem Intervallsystem arbeiten. Es hat eigentlich nicht sehr viel Sinn, einen der indischen Musikphilosophie entstammenden Raga Todi in unser Intervallsystem zu pressen. Waveform sollte in einer späteren Softwareerweiterung auch die unterschiedlichen Intonationen mit berücksichtigen. Dann wäre das System perfekt.

Zwanzig der erstellten Keyboardlayouts lassen sich zu einer sogenannten Scratch-Page zusammenfassen. Ein Preset mit 20 verschiedenen Keyboard-Belegungen gewissermaßen. Aus diesem Scratch-Page können wir dann ohne Diskettenoperationen einzelne Layouts abrufen, studieren beziehungsweise spielen. Diese Zusammenstellungen sind natürlich editierbar.

Musicalc ist ein Programmsystem für den Anspruchsvollen unter den Commodore 64-Musikern. Da man jeweils nur eine Stimme auf der Tastatur spielen kann, eignet es sich nicht zum polyphonen Spiel. Es ist mehr als Kompositions- und Abspielsystem gedacht. Bisher ein Novum: Es unterstützt den, der seinen Computer zusammen mit anderen Rhythmusinstrumenten beziehungsweise einem oder mehrerer weiterer C 64 synchron betreiben will. Sollte demnächst auch der Notenausdruck funktionieren, ist Musicalc in meinen Augen, momentan jedenfalls, eines der bestdurchdachten Musik-Software-Pakete.

(Richard Aicher/aa)

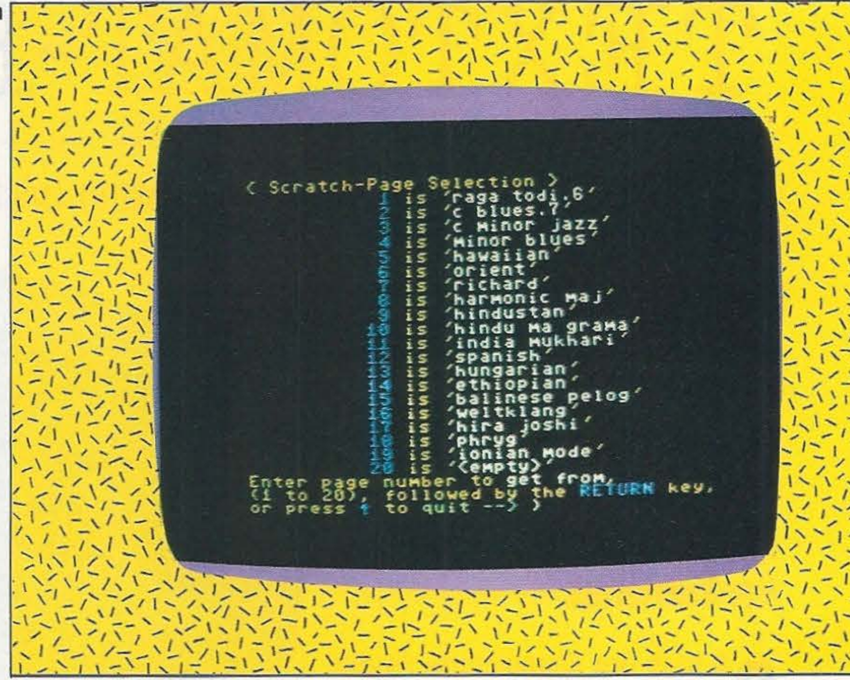


Bild 9. In diesen Tonleitern kann man mit Keyboardmaker schweigen

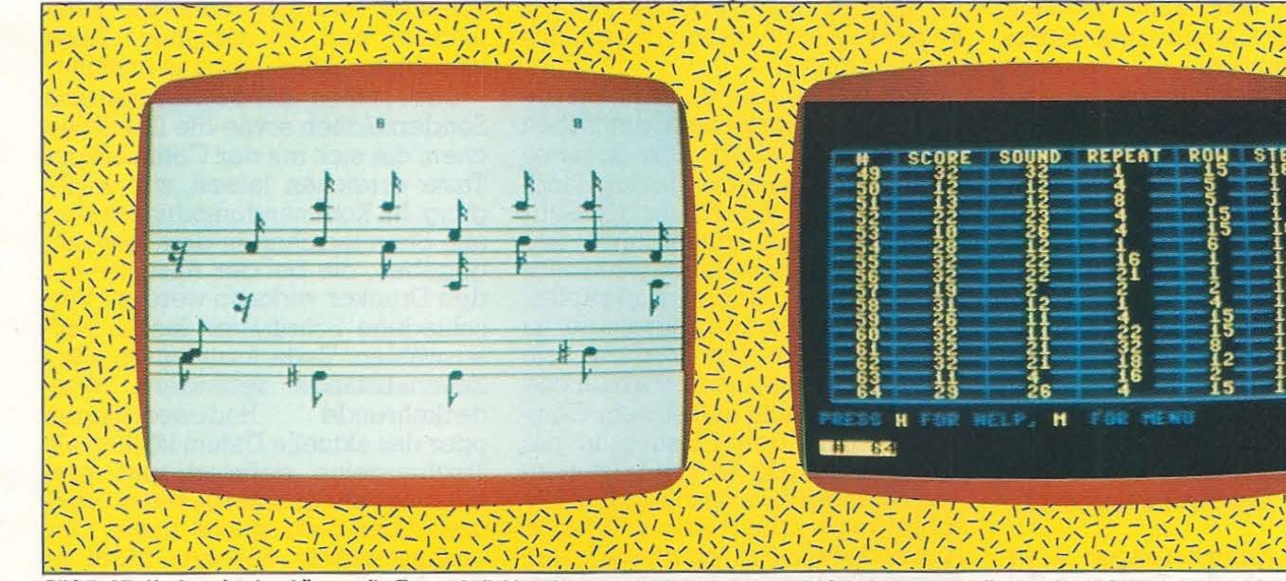


Bild 7. Mit Keyboardmaker können die Tasten beliebig mit Tönen belegt werden

Bild 8. Das Keyboardlayout des indischen Raga Todi. Kein Problem für Keyboardmaker.

Info: Musicalc ist in Deutschland bei Lucius Computer-Programme, Theodor-Körner-Str. 5, 4220 Dinslaken 1, Tel. 02134/52782 zu erhalten.

TEXTOMAT-Büroanwen

Gute Textverarbeitungs-Programme kosten häufig mehrere hundert Mark. Aber es geht auch billiger. Beim Textomat von Data Becker stimmt der Preis, und er kann auch mit den Leistungen wesentlich teuerer Programme konkurrieren.

Bild 1. Die Menüstruktur des Textomats

Oft muß der Commodore 64 die Aufgaben eines Personal Computers übernehmen. So erfreut er sich beim Einsatz in der Textverarbeitung zunehmender Beliebtheit. Einen wesentlichen Beitrag dazu haben die Entwickler spezieller Textverarbeitungsprogramme, wie Textomat, geleistet. Die Textverarbeitung, eine der sinnvollsten und nützlichsten Anwendungen des Computers, birgt aber ein Handicap in sich: Sie läßt kaum Kompromisse zu.

Sinnvolles Arbeiten ist mit solchen Programmen nur dann möglich, wenn einige Grundvoraussetzungen erfüllt werden:

— Ein deutscher Zeichensatz auf dem Bildschirm, leicht zu handhabende, deutsche Benutzerführung
— Umfangreiche Textformatierungsbefehle für den Ausdruck und den Bildschirm
— Die Möglichkeit zur Diskettenverwaltung
— Umfangreiche Anpassungsmöglichkeiten an verschiedene Drucker.
— Verschiebe,- Ersetz,- und Kopierfunktionen.
— und möglichst eine Schnittstelle zu einem Datenverwaltungsprogramm.

Alle diese Anforderungen zu erfüllen, verspricht der Textomat. Seine Geschichte ist eigentlich einen eigenen Artikel wert, denn angeboten wird das Programm schon seit über einem Jahr. In dieser Zeit reifte der Textomat von Version zu Version — nicht immer zur Freude der Käufer. Zum Test stand aber die bislang neueste Auflage dieses Programms zur Verfügung.

Textomat wird komplett mit Diskette und umfangreichem deutschen Handbuch geliefert. Das gesamte Programm ist ebenso wie das Handbuch ganz in Deutsch verfaßt. Sehr erfreulich, denn immer noch geistert in manchen Softwarehäusern die Vorstellung herum, daß professionelle Anwenderprogramme in englischer Sprache verfaßt sein müssen. Das insgesamt vorbildliche Handbuch macht selbst dem Computerneuling den Einstieg in die Textverarbeitung leicht. Es ist in einen Übungsteil, in dem alle Befehle schrittweise erklärt sind und einen Anwenderteil, der nochmals genauer auf alle Befehle eingeht, gegliedert.

Der Weg, der mit dem Textomat bei der Art der Textbehandlung beschritten wurde, hebt sich in einigen Punkten von anderen Konkurrenzprodukten ab. Der Textomat unterscheidet zwischen Schreib-, Kommando- und Menümodus. Texte werden im Schreibmodus erstellt.

Dazu stehen alle Buchstaben und Sonderzeichen sowie die Grafikzeichen, die sich mit der Commodore-Taste erreichen lassen, zur Verfügung. Im Kommandomodus werden alle Steuerzeichen in den Text eingegeben, die bei der Ausgabe auf den Drucker wirksam werden. Verschiedene Schriftarten lassen sich auswählen, Texte formatieren oder Zeilenabstände verändern. Wiederkehrende Redewendungen oder das aktuelle Datum können als Textbausteine definiert und problemlos aneinandergekettet werden; aber dazu später mehr. Der umfangreichste der drei Modi ist der »Menümodus«. In der untersten Bildschirmzeile sind die vier Hauptmenüs »Edit, Formular Ausgabe und Dienst« abgebildet. Nach dem Anwählen eines dieser Punkte beginnt

TEXTOMAT-Büroanwen

dung zum kleinen Preis

der Kampf durch die Menülandschaft, die aus bis zu drei Untermenüs besteht (Bild 1). Soll beispielsweise der Inhalt einer Diskette eingelesen werden, so ist zuerst das Dienstmenü zu wählen, von da aus der Unterpunkt »Floppy« und dort angelangt wiederum der Punkt »Disketteninhalt zeigen«. Leider führt der Weg durch die Menüs nicht nur von oben nach unten, sondern auch von unten nach oben.

Eines der größten Handicaps des C 64 wurde beim Textomat elegant umgangen. Eine Textverarbeitung fordert in der Regel eine Darstellung von 80 Zeichen pro Zeile auf dem Bildschirm, denn ein Ausdruck hat ja meistens auch dieses Format.

Bild 2. Das Texteingabefeld

Da der C 64 aber nur mit einer 80-Zeichen-Karte oder mit einem entsprechenden Programm 80 Zeichen darstellen kann (wobei die Leserlichkeit bei der Software-Methode in Verbindung mit einem Fernseher stark leidet), wurden die Zeichen in ihrer Originalgröße belassen. Die Darstellung von mehr als 40 Zeichen wird durch automatisches horizontales Verschieben des Textes bewerkstelligt. Der Text kann dann ohne Rücksicht auf Textformatierung als zusammenhängende Folge von Wörtern eingegeben werden. Leider werden am definierten Zeilenende die Wörter unterbrochen und in der nächsten Zeile fortgesetzt. Diese Tatsache stört etwas, kann aber mit dem Prinzip des bildschirmorientierten Aufbaus erklärt werden. Die notwendige Formatierung des Textes wird erst kurz vor dem Ausdruck vorgenommen. Der Vorteil dieses Konzepts liegt darin, daß Texte allen Eventualitäten angepaßt werden können. Dazu stellt der Textomat eine nützliche Hilfsfunktion bereit. Alle häufig gebrauchten Formatbefehle können in Form von Formularen zusammengefaßt und abgespeichert werden. Ein einziger Text kann so durch einfaches Nachladen der Formatbefehle ein unterschiedliches Bild annehmen.

Wer aber lieber den Text schon auf dem Bildschirm formatiert haben will, kann dies durch Verwendung von »Fest-Zeichen« erreichen (Bild 2). Nun zur Praxis. Die erste Überraschung erlebt der Anwender kurz nach dem Laden. Er wird gefragt, welchen Schrifttyp er für die Bildschirmdarstellung wählt. Zur Auswahl stehen Standard-Deutsch/Amerikanisch und Alt-deutsch/Amerikanisch, sowie ein individueller Zeichensatz. Der Unterschied zwischen den Standard- und den Alt-Zeichensätzen ist lediglich eine etwas verschnörkelte Darstellung auf dem Bildschirm (nicht auf dem Drucker!). Eine hübsche, aber nicht notwendige Funktion. Interessant ist die individuelle Zeichendarstellung. Mit ihr ist es möglich, besondere wissenschaftliche oder sogar japanische Zeichen auf dem Bildschirm darzustellen (Bild 3). Sinnvoll anwendbar ist diese Funktion allerdings nur dann, wenn gleichzeitig ein Drucker mit ladbarem Zeichensatz, wie der Epson FX-80, zur Verfügung steht. Erst dann werden die auf dem Bildschirm dargestellten Sonderzeichen auch ausgedruckt. Ansonsten bleibt die Wahl des Zeichensatzes eine hübsche Spielerei. Leider wird im Handbuch auf diese Funktion nicht genügend eingegangen. Beim Test gelang es deshalb erst nach mehreren Stunden, einen neuen Zeichensatz auszudrucken. Im einzelnen ist es notwendig, einen Zeichensatz zu entwerfen — ohne geeignetes Hilfsprogramm ein fast aussichtsloses Unterfangen. Außerdem muß der gesamte Vorgang für den Drucker wiederholt werden, denn Bildschirmdarstellung und Druckbild (Bild 4) sind zwei grundverschiedene Angelegenheiten. Der fertige Zeichensatz wurde dann vor den Textomat geladen und in den Zeichenspeicher des FX-80 übertragen.

Der Textomat kann auch den deutschen Zeichensatz darstellen. Nach der Wahl des deutschen Zeichensatzes werden einige Tasten des C 64 umbelegt, das »Z« und das »Y« werden vertauscht und die deutschen Umlaute definiert. Der erste Versuch, diese Zeichen auch auszudrucken, wurde mit dem Epson-Drucker vorgenommen, der ja bekanntlich über einen deutschen Zeichensatz verfügt. Leider ist es vor-

dung zum kleinen Preis

TEXTOMAT-Büroanwen

her notwendig, jedem einzelnen Zeichen einen eigenen, im Druckerhandbuch beschriebenen, Wert zuzuordnen; ein langwieriges und fehlerträchtiges Unterfangen. Andererseits wird dadurch sichergestellt, daß größte Flexibilität erhalten bleibt. Die fertige Einstellung wird zusammen mit den ebenfalls definierbaren Steuercodes auf einer sogenannten Parameterdiskette abgespeichert.

Dadurch entfällt die erneute Eingabe aller Werte. Wünschenswert wäre allerdings, wenn eine solche Einstellung für die gängigen Schnittstellen und Drucker bereits auf der Diskette mitgeliefert würde. Denn nicht jeder Anwender ist mit den innersten Geheimnissen der Zeichendarstellung vertraut.

Zurück zum deutschen Zeichensatz. Nach den vorgenommenen Einstellungen druckte der Textomat mit einem Epson FX-80 ordnungsgemäß den gesamten Zeichensatz aus. Lediglich beim Görlitz-Interface älterer Bauart kam es zu gar keinem Ausdruck, da erst die neueren Versionen mit Textomat zusammenarbeiten. Wie verhielten sich aber Commodore-Drucker, namentlich der 1526 oder MPS 802, die ja keinen deutschen Zeichensatz kennen? Das Ergebnis war verblüffend und erfreulich zugleich, der angeschlossene MPS 802 stoppte bei jedem Umlaut kurz, druckte dann aber das gewünschte Zeichen. Ein kleiner Programmiertrick mit großer Wirkung! Das beim MPS 802 (früher VC 1526) mögliche, eine frei definierbare Zeichen wird durch ständiges Umdefinieren zur Darstellung der Umlaute verwendet. Auch die grafikfähigen Commodore-Drucker wie der MPS 801 (VC 1525) ließen keinen Umlaut aus. Kein anderes uns bekanntes Textverarbeitungssystem verfügt über diese Besonderheit. Der Ausdruck dauert zwar etwas länger, aber er erfüllt alle Anforderungen an einen sauberen Ausdruck. Nur wer ganz genau hinschaut, erkennt, daß die Umlaute nicht exakt zwischen den anderen Buchstaben stehen.

Einer der Hauptvorteile der computergestützten Textverarbeitung ist die Möglichkeit, den Text beliebig zu verschieben, zu editieren und zu korrigieren, ohne dabei Zeit zu verlieren und kiloweise Altpapier zu produzieren. Der Textomat ist in diesem Bereich mit einigen sehr interessanten Befehlen ausgestattet. Die sogenannten Blockoperationen erlauben es, einzelne Textblöcke zu markieren und diese nach Belieben zu verschieben, zu löschen, oder zu kopieren.

Leider muß ein Block immer nur aus ganzen Zeilen bestehen, so daß nach dem Blockkommando immer noch etwas nachgebessert werden muß. Aber auch hierzu sind eine Reihe von Tastenfunktionen vorgesehen. Wer den Commodore-Editor gewöhnt ist, muß allerdings eine kleine Änderung berücksichtigen. Bei einem Druck auf die DEL-Taste wird das Zeichen gelöscht, an dem der Cursor steht und nicht wie üblich, das Zeichen links von ihm. Nach Betätigen der Taste »INS« wird in der Informationszeile am oberen Bildschirmrand »Einf.« für »Einfügen« angezeigt. Alle nachfolgend eingegebenen Zeichen werden nun in den bereits bestehenden Text eingeschoben.

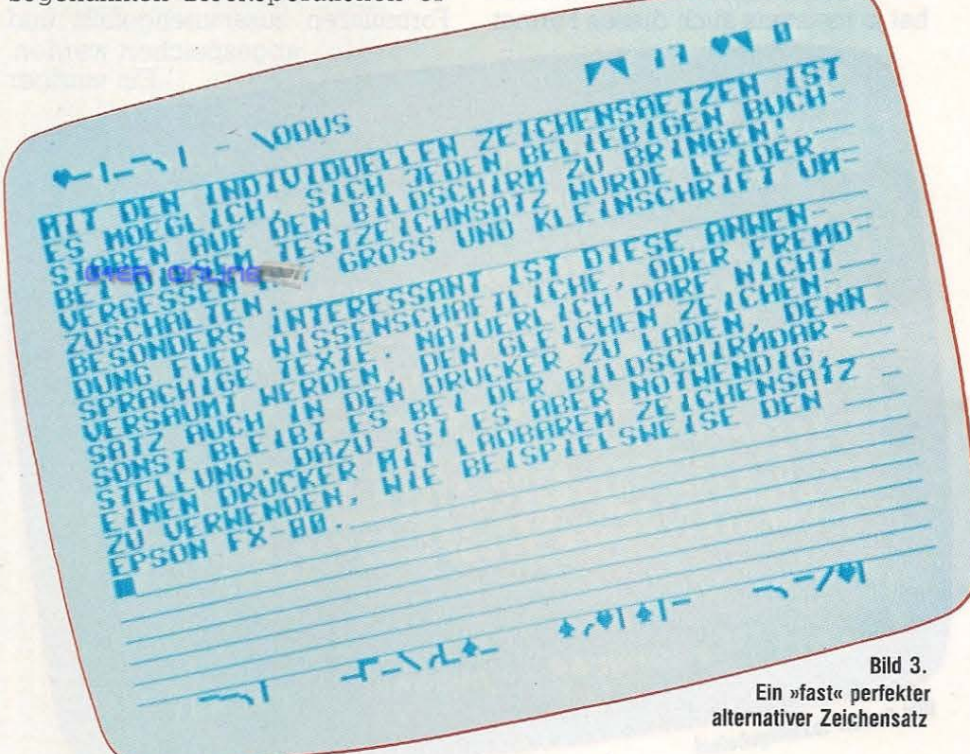


Bild 3. Ein »fast« perfekter alternativer Zeichensatz

Die Funktionstasten f3 bis f6 machen ein wortweises Vor- und Zurückspringen, aber auch Sprünge an den Textanfang und das Textende möglich.

Eigentlich wäre der Textomat jetzt schon ein gutes Textverarbeitungsprogramm, aber er bietet noch mehr. Einfache arithmetische Operationen sind ihm nichts Unbekanntes. Im Kontrollmodus können beliebig viele Zahlen zur Berechnung herangezogen werden. Die Genauigkeit ist hinreichend, denn sogar

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789
DIES IST EIN BEISPIEL FUER EINEN
ALTERNATIVEN ZEICHENSATZ. DAZU IST
ES ALLERDINGS NOTWENDIG, EINEN
DRUCKER MIT LADBAREM ZEICHENSATZ ZU
BESITZEN. DER VORTEIL DES TEXTOMAT
BESTEHT DARIN, DIESEN ZEICHENSATZ
AUCH AUF DEM BILDSCHIRM DARSTELLEN
ZU KOENNEN. EIN WESENTLICHER VORTEIL
BEI WISSENSCHAFTLICHEN ARBEITEN
```

Bild 4. Probeausdruck eines alternativen Zeichensatzes

dung zum kleinen Preis

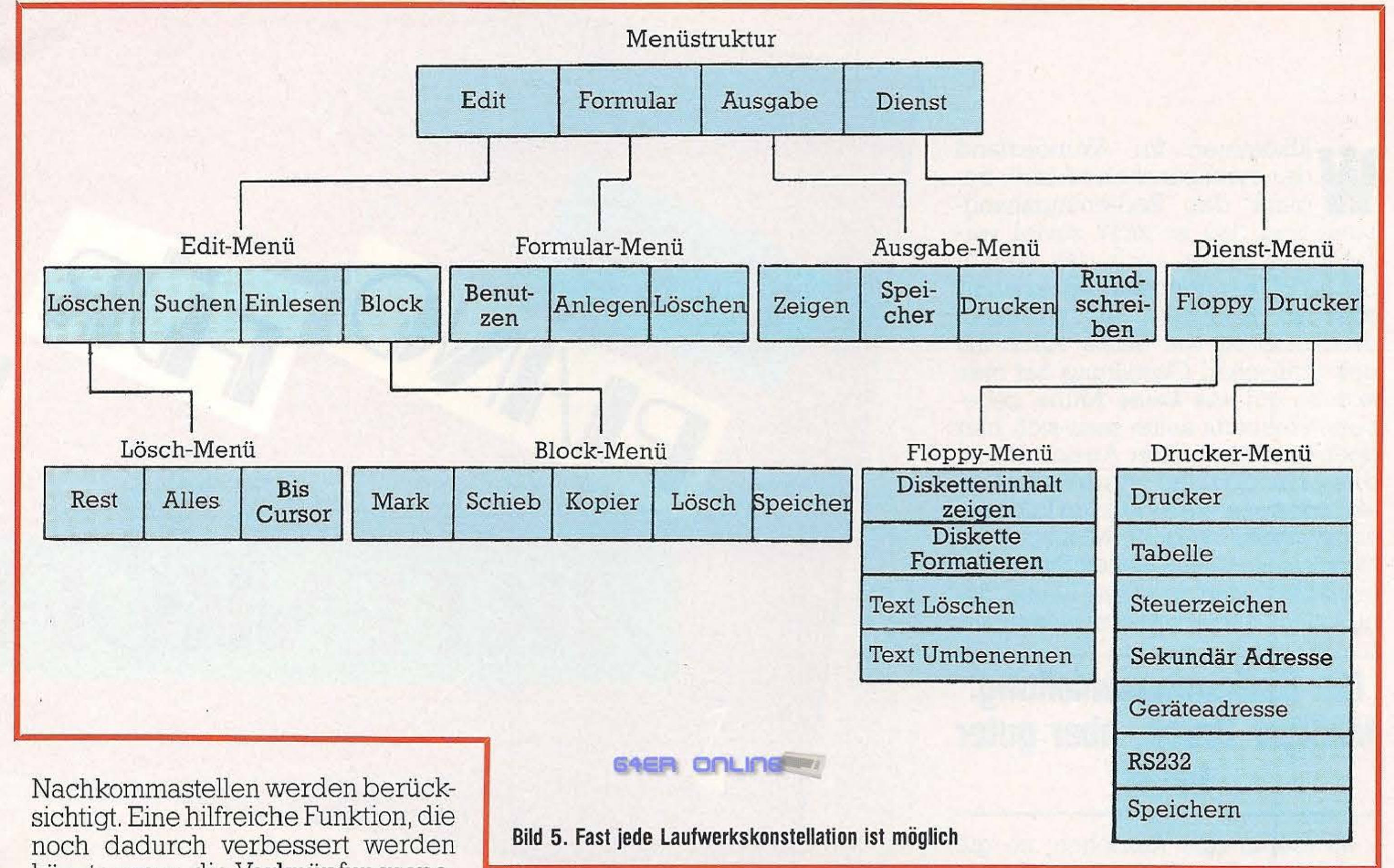


Bild 5. Fast jede Laufwerkskonstellation ist möglich

Nachkommastellen werden berücksichtigt. Eine hilfreiche Funktion, die noch dadurch verbessert werden könnte, wenn die Verknüpfungsoperanten definierbar wären.

Wesentlich wichtiger, da häufiger gebraucht, ist die Schnittstelle zu einem Datenverwaltungsprogramm. Der Sinn dieser Funktion liegt in dem Erstellen von Formbriefen und Rundschreiben. Im Handbuch wird auf das Datenverwaltungsprogramm aus gleichem Hause verwiesen, dem Datamat. Leider konnten wir diese Funktion nicht in befriedigendem Maße testen, da uns dieses Programm nur in einer veralteten, viel zu langsamen Version vorlag. Mit etwas Übung und Geduld kann es aber gelingen, den richtigen Datensatz aus dem entsprechenden Datenfeld zu lesen. Sie merken schon, ganz einfach ist diese Funktion nicht zu handhaben.

Wer glaubt, jetzt das Ende der Fähigkeiten von Textomat erreicht zu haben, irrt. Beim Test gefielen noch zwei Dinge besonders. Die Diskettenoperationen und die Suchfunktion. Erstere beginnen mit der Wahl des entsprechenden Laufwerkskonstellation (Bild 5). In einem einfachen Wahlmenü kann eingestellt werden, wieviele Laufwerke (1 oder 2) und mit welchem Interface (IEEE 488 oder serieller IEC-Bus) sie angeschlossen sind. Der Einsatz zweier 1541 oder einem 4040 Doppellaufwerk wird somit unterstützt. Natürlich sind auch alle Kommandobefehle des Laufwerkes verfügbar. Die Besitzer eines Kassettenlaufwerkes werden allerdings enttäuscht. Denn weder das Programm noch seine Dateien arbeiten mit der Kassette zusammen.

»Suchen und Ersetzen« ist die Aufgabe der Suchfunktion. Beliebige Textstrings können hiermit entweder einzeln oder global durch einen anderen ersetzt werden. Dies ist ein anderer Weg, Serienbriefe zu erzeugen, indem der Name eines Adressaten durch den eines anderen im gesamten Text ersetzt wird. Das ist zwar etwas umständlicher, aber es erspart den Kauf des Datamat (jedenfalls in dieser Beziehung).

Der Textomat stellt ein leistungfähiges Textverarbeitungsprogramm mit leichten Verbesserungsmöglichkeiten dar. Mit ihm ist ein durchaus sinnvolles und komfortables Arbeiten in vielen Bereichen möglich, auch wenn die Steuerung der vielen Funktionen manchmal etwas gewöhnungsbedürftig ist. Eine einzige Funktion konnte im Test leider gar nicht überzeugen: Das Ausdrucken. Nach dem Start des Ausdrucks ist kein Anhalten und keine Korrektur mehr möglich. Dies ist besonders widrig, wenn das gesamte System noch nicht optimal ist. Wer vergessen hat, seine Drucker anzustellen oder wer in Verbindung mit einem Epson-Drucker vergessen hat, die Parameterdiskette zu komplettieren, wird nichts erhalten. Auch nicht seinen Text zurück. Deshalb Vorsicht beim Ausdruck, erst abspeichern, dann drucken! Es wäre schön, wenn die Anpassung an die verschiedenen Drucker etwas komfortabler gestaltet werden könnte, denn die Umrechnung und die Eingabe von hexadezimalen Zahlen ist nicht jedermanns Sache.

Einen Bereich gibt es allerdings, in dem der Textomat »noch« nicht zu schlagen ist: Der Textomat ist jede seiner 99 Mark wert und es wäre wünschenswert, wenn auch andere Softwarehäuser ähnlich gute Programme in dieser Preisklasse anbieten würden.

(Arnd Wängler/aa)



Willkommen im »Wunderland der synthetischen Musik«, beginnt das Bedienungshandbuch und das ist nicht zuviel versprochen. Aber auch das »Handbuch« ist ein Wunder, es erinnert eher an Fotokopie, als an Buchdruckerkunst von heute. Auch mit der grafischen Gestaltung hat man sich so gut wie keine Mühe gegeben. Vielleicht sollte man sich hier doch endlich von der Ansicht lösen, diese Handbücher würden nur Freaks lesen, denen es nur um Informationsvermittlung geht. Musiker, und sicher viele davon wollen Synthimat einsetzen, sind kreative Leute, denen meist auch die Form wichtig ist.

Die Bedienungsanleitung: Mäßiger Druck, aber guter Inhalt

So mäßig das Aussehen, so gut der Inhalt. Der Autor hat sich viel Mühe gegeben und selbst schwierige Sachverhalte anschaulich erklärt. So erfährt man nicht nur alles wesentliche zu den einzelnen Bedienungsbefehlen, sondern auch das Wichtigste über Klangsynthese, Filterung, Hüllkurven, Schwingungsformen, Funktionsweise des SIDs und so weiter.

Das Programm bietet Möglichkeiten, von denen manch »echter« Synthesizer nur träumt. Insgesamt stellt es elf Oszillatoren zur Verfügung. Zwei getrennt spielbare »Tastaturen« erlauben, Begleitung und Melodiestimme mit unterschiedlichen Sounds unabhängig voneinander, zu spielen. Bis zu 256 verschiedene Klänge merkt sich Synthimat 64 softwaremäßig. Ein komplettes Lied kann in Realtime auf Diskette gespielt werden, als wäre die Diskette ein Tonband. Insgesamt finden auf einer Diskette 9x256 Soundprogramme und neun verschiedene Lieder Platz.

Das Programm schöpft alle Möglichkeiten des SIDs voll aus. Synthimat 64 stellt alle drei DCOs des SID-Chips, mit allen acht möglichen Kurvenformen zur Verfügung. Die DCOs lassen sich einzeln in ⅛ Ton-Schritten stimmen. Auch alle Filterparameter, die der Chip bietet, stehen zur Verfügung. Gleiches gilt für die ADSRs. Sämtliche Parameter lassen sich für jede der drei Stimmen separat einstellen.

Elf Oszillatoren

Zusätzlich zu den Features des SIDs, realisierten die Entwickler softwaremäßig noch 8 LFOs. Diese verfügen genau wie die Oszillatoren über 8 verschiedene Wellenformen. Jeweils 2 LFOs modulieren bei Bedarf einen DCO, und zwar einer die Pitch, der andere die Pulsbreite. Von den 2 verbleibenden LFOs kann man mit einem den Filter, mit dem zweiten die Lautstärke der drei Stimmen beeinflussen. Die Geschwindigkeit jedes LFOs läßt sich separat bestimmen.

Recht komfortabel gerieten auch die Ringmodulations- und Syncmöglichkeiten. Acht verschiedene Arten der Beeinflussung der DCOs untereinander stehen zur Auswahl. Zu guter letzt hat man auch einen Pitchbender nicht vergessen.

An Möglichkeiten mangelt es Synthimat sicher nicht.

Der wunde Punkt am Ganzen, zumindest für jeden klaviergewohnten Musiker, ist, wie üblich, die Tastatur selbst. Es erfordert schon etwas Umgewöhnungszeit und Geduld, bis man auf den Schreibmaschinentasten annähernd so flüssig spielen kann wie auf einer Klaviertastatur. Vor allem bei beidhändigem Spiel auf den übereinanderliegenden Tastaturbereichen, kann es schon einmal passieren, daß man sich die Finger verknotet. Doch mittlerweile gibt es hier Abhilfe. Zwei ordentliche Klaviertastaturen kamen in jüngster Zeit auf den Markt. Angeschlossen an den Commodore 64 wird dieser dann wirklich zum Musikinstrument.

Nun soll es aber losgehen. Wir legen die Diskette ein und tippen auf der Tastatur des Commodore 64:

Ein Konzert für drei Stimmen, live gespielt auf einer Schreibmaschinentastatur — aufgenommen mit einer zum Tonbandgerät umfunktionierten Diskettenstation. Das Programm Synthimat 64 macht's möglich.



Bild 1. Synthimat-64-Eingabefelder auf dem Bildschirm

LOAD "*",8,1 dann RUN. Der Ladevorgang startet, und nach kurzer Zeit erscheint ein farbiges Bild auf dem Schirm (Bild 1).

Spätestens jetzt hat der Computer vergessen, daß er ein Computer ist. Man sollte ihn ab sofort auch besser »Synthi« nennen. Alle Tastenfunktionen sind jetzt neu definiert und speziell auf das Programm zugeschnitten. Dies wird höchstens den Computerfan anfangs irritieren, der Musiker kann so jedoch getrost vergessen, daß er es mit einem Computer zu tun hat.

Getrennte Keyboards, für Solo und Begleitung

Drückt man jetzt auf eine Taste der oberen zwei Buchstabenreihen, so klingt ein Ton aus dem Lautsprecher. Wir spielen gerade auf dem sogenannten Solokeyboard. Die dritte und vierte Tastenreihe stellen die zweite Klaviatur, das Begleitkeyboard dar. Drei Töne können wir maximal gleichzeitig auf den zwei Tastaturen spielen. Bei beiden Klaviaturen entspricht jeweils die obere Tastenreihe den schwarzen Halbtontasten, die untere, den weißen Ganztontasten einer »richtigen« Klaviertastatur.

Einen gespielten Ton meldet Synthimat 64 grafisch mittels weißem Punkt beziehungsweise Balken, auf der entsprechend symbolisierten Taste am Bildschirm. Drückt man auf die Shift-Taste, so schreibt Synthimat 64 die Buchstabenbezeichnung der Commodore 64-Tastatur in die Tastengrafik des Bildschirms.

Betrachten wir das Arbeitspanel unseres Synthimat 64 (Bild 1). Hier sind die einzelnen Funktionsbereiche des Schirmbildes zu sehen. Zusammengehörige erkennt man jeweils leicht an der gemeinsamen Farbe. Diese Bildschirmgrafik bleibt übrigens während des gesamten Arbeitens mit Synthimat 64 erhalten. Sie bildet sämtliche veränderbare Funktionen mit den jeweils eingestellten Werten ab. Mühsames Hin- und Herschalten diverser Menüs bleibt also erspart.

Der obere, von rotbraunen Linien eingeschlossene Teil des Bildschirms, zeigt die für die Tonerzeugung wichtigen Parameter. Im unteren Bildschirmbereich erkennen wir die beiden Tastaturdiagramme, die Anzeigenfelder für die Stimmung der Oszillatoren und die aktuelle Bedienungsfunktion.

Bewegung bringen zwei Anzeigen ins Bild. Eine Uhr (unter der weißen Schrift Synthimat 64), die auf Ortszeit gestellt werden kann sowie die SYS-Uhr im rechten unteren Bildabschnitt. Letztere zeigt an, wie schnell das Programm gerade abläuft.

Wie arbeitet man nun mit Synthimat? Zunächst will die Software wissen, was man überhaupt mit ihr anzustellen gedenkt. Sounds einstellen, die VCOs den Keyboards zuordnen, Songs aufnehmen oder abspielen, Glides und Tunes einstellen, oder Sound und Songprogramme von der Diskette in den Computer laden beziehungsweise speichern.

13 verschiedene Bedienfunktionen bringen Ordnung in das System

Insgesamt warten 13 verschiedene Bedienungsfunktionen auf ihren Einsatz. Die jeweils gültige zeigt Synthimat 64 mit weißer Schrift auf dem Bildschirm, oberhalb der Melodietastatur, an. Die einzelnen Funktionen rufen wir mit den vier Funktionstasten auf. f1 dient zum Durchblättern der einzelnen Funktionen, f3 startet die jeweilige Funktion und mit f7 verläßt man sie wieder.

Betrachten wir kurz die einzelnen Bedienungsfunktionen. Mit SET REAL-TIME CLOCK stellen wir die eingebaute Echtzeituhr. Mit der Funktionstaste f3 wählen wir den einzustellenden Bereich (Std., Min., Sec., $\frac{1}{10}$ Sec.); jeder Druck auf f3 schaltet dann automatisch einen Bereich weiter. Dieses praktische »Step-Prinzip« wird für alle wählbaren Einstellungen beibehalten. Mit f5 läßt sich nun der gewünschte Wert einstellen. Ein Druck auf f7 und

die Uhr startet. Damit hat man die Funktion SET REAL-TIME CLOCK wieder verlassen. Wir können nun einen anderen Funktionsbereich anwählen.

Mit der Funktion SET VCOS TO KEYBOARD ordnen wir die drei Oszillatoren den beiden Keyboards zu. Jede der drei Stimmen, die jeweils aus einem Oszillator mit zugehöriger Klangeinstellung bestehen, läßt sich entweder auf das Solo- oder das Begleitkeyboard legen. Die betreffenden Stimmen werden sofort durch inverse Darstellung der entsprechenden Ziffern 1, 2 oder 3, links neben den Keyboards, am Bildschirm gekennzeichnet.

Während das Einstellens kann man übrigens weiterspielen. Man hört bei allen Einstellarbeiten sofort die Auswirkung einer Parameteränderung auf den Sound. Auch im Schirmbild zeigt Synthimat 64 jede Parameteränderung sofort grafisch an. Entweder numerisch oder mittels Hintergrundeinfärbungen beziehungsweise mit Symbolen. Diese Methode wird im ganzen Programm beibehalten und vereinfacht das Arbeiten ungemein. Deshalb kann man den Bildschirm wirklich als das Bedienfeld eines komfortablen Synthesizers betrachten, der ständig alle gültigen Werte und Zuordnungen anzeigt.

Will man mit anderen Instrumenten zusammenspielen, müßen wir Synthimat 64 erst einmal stimmen. Dies geht mit der Funktion SET TUNE FOR VCO. Die Stimmung kann in 1/8 Tonschritten über den Bereich einer Oktave verändert werden. Da sich die einzelnen Oszillatoren getrennt stimmen lassen, lassen sich auf diese Weise Phasingklänge erzielen (Schwebung). Dies ist wichtig für Streichereinstellungen. Hat man sich jedoch beim Durchsteppen der Tuningeinstellungen einmal um einen Step geirrt, müssen alle nachfolgenden 99 Werte brav durchschritten werden. Rückwärts schreiten ist leider nicht möglich. Das Tuning der drei Oszillatoren wird auf dem Schirm numerisch angezeigt, ganz rechts unten.

SET GLIDE TO VCO ermöglicht gleitenden Tonverlauf zwischen zwei aufeinanderfolgend gespielten Tönen. Musiker nennen diesen Effekt auch Portamento. Die Glide-Zeiten können für jede Stimme unabhängig eingestellt werden. Die kleinste Abstufung ist jedoch bereits etwas groß gewählt.

EQUALIZE VCOS BY erlaubt das Kopieren sämtlicher Parametereinstellungen einer Stimme auf eine andere, mit einem einzigen Befehl. Das erspart viel Zeit und wird vor allem benötigt, wenn Akkorde gespielt werden sollen, deren Töne gleich klingen.

SOLO/MULTI PLAY MODE: In der Solostellung spielt nur ein VCO jeweils einen Ton, im Multiplaymode hingegen spielen alle verfügbaren Oszillatoren den gleichen Ton. Bei harmonisch verstimmten Oszillatoren erklingt so mittels einem Tastendruck ein ganzer Akkord auf einmal.

Mit der ACCOM: MELODY-CHORD-Funktion können im Chord-Modus auf der Begleittastatur Arpeggios gespielt werden. Drückt man zwei oder mehrere Tasten gleichzeitig, wird — angefangen mit dem tiefsten zum höchsten — zyklisch ein Ton nach dem anderen durchgespielt. Ähnlich der Begleitautomatik älterer Heimorgeln.

Die Floppy 1541 als Tonbandgerät

Zum Tonbandgerät wird die Floppy in den beiden Bedienfunktionen ENGAGE DIRECT TO DISK und ENGAGE DIRECT FROM DISK. Im ersten Modus, dem Aufnahmemodus, wird alles was man auf der Tastatur spielt, direkt auf die laufende Diskette gespeichert. Das Ganze ähnelt deshalb einer Tonbandaufnahme, mit dem Unterschied, daß hier digitale Daten aufgenommen werden. Hat man sich einmal verspielt, was bei der Schreibmaschinen-Tastatur leicht vorkommen kann, muß man noch einmal von vorne beginnen. Einzelne Teile des Stückes lassen sich nicht löschen und neu einspielen. Synthimat 64 ist das einzige Programm, das Songs auf diese Art und Weise direkt auf die laufende Diskette speichert, ohne Umweg über den Arbeitsspeicher des Computers.

Die Sequenzen können nur in Realtime über das »Keyboard« eingespielt werden. Leider existiert keine Composerfunktion für weniger virtuose Softwarefreaks.

Insgesamt haben auf einer Diskette 9 Lieder Platz. Sie werden als sogenannte Soundfiles gespeichert. Welche Soundfilenummer ein Song bekommen soll, können wir in der Bedienungsfunktion SET DISK FILE bestimmen.

Neben den Song-Directfiles existieren noch sogenannte Registerfiles. In diesen legt man die Klangparameter ab. Insgesamt haben in jedem Registerfile 256 verschiedene Klangeinstellungen, also komplette Synthiprogrammierungen, Platz.

256 verschiedene Klangprogramme im Arbeitsspeicher

Recht einfach geht die Soundprogrammierung in der Bedienfunktion SET SID 6581 REGISTER, vonstatten. Insgesamt existieren 44 voneinander unabhängig einstellbare Klangparameter (Bild 2).

Mit den zwei Funktionstasten f1 (vorwärts) und f2 (rückwärts) dirigiert man einen Cursorpfeil von Parameter zu Parameter. Am gewünschten Parameter angelangt, erhöht ein Druck auf Taste f5 den Parameter um einen Wert. Die Taste RETURN erniedrigt denselben jeweils um einen Wert. In Bild 2 erkennen wir, welche Parameter im einzelnen veränderbar sind und in welchen Abstufungen. Es lassen sich dann in jedem Feld die möglichen Werte in aufsteigender oder abfallender Reihenfolge durchschreiten. Die Klangveränderung hört man sofort. Es handelt sich also um einen echten Editmodus.

Synthimat stellt alle drei DCOs des SID-Chips, inklusive aller acht möglichen Kurvenformen, zur Verfügung. Man nennt die eigentlich digital gesteuerten Oszillatoren hier, aus historischen Gründen, VCOs. Die jeweils angewählte Kurvenform erscheint, im zugehörigen VCO-Anzeigefeld, in Zeile 1 unseres Bildschirms. Die Kurvenformen werden mit Symbolen angezeigt. Direkt daneben die Fußlage des jeweiligen Oszillators. Fußlagen sind Oktavbezeichnungen der Orgelkunde. Die tiefste hier spielbare Fußlage ist 64 Fuß, die höchste 1 Fuß. Insgesamt stehen 8 Oktaven zur Verfügung. Hat man die Rechteckform für einen Oszillator angewählt, kann man auch dessen Pulsbreite variieren, angezeigt neben der Fußlage. Die Pulsbreite kann in 4096 Schritten verändert werden. Sämtliche Parameter lassen sich für jede der drei Stimmen separat einstellen.

Zusätzlich zu 3 VCOs des SIDs realisierten die Entwickler softwaremäßig noch 8 langsamschwingende Oszillatoren (LFOs). Diese verfügen, genau wie die Oszillatoren, über 8 verschiedene Wellenformen. Je-

weils 2 LFOs modulieren einen DCO und zwar einer die Pitch, der andere die Pulsbreite. Ihre Parameter stehen in der zweiten Zeile des Bildschirms, jeweils unter dem zugehörigen VCO-Einstellfeld. Die ausgewählte Kurvenform wird mit einem Symbol, die Modulationsfrequenz numerisch angezeigt. Die Modulationstiefe kann nur für alle drei Oszillatoren gemeinsam bestimmt werden und zwar mit dem roten Balken unter der Realtime-Uhr. Von den 2 verbleibenden LFOs kann man mit einem Filter, mit dem zweiten die Lautstärke der drei Stimmen beeinflussen. Mit der Zuordnung von Kurvenformen und LFOs muß man etwas experimentieren. Die Geschwindigkeit jedes LFOs läßt sich separat bestimmen. LFO 7 wirkt, wie schon gesagt, auf den Filter, LFO 8 auf den DCA.

Die Hüllkurvengeneratoren (ADSRs), bestimmen den zeitlichen Verlauf der Lautstärke des Klanges. Bei Synthimat 64 ist pro Stimme ein ADSR vorhanden. Alle vier Hüllkurvenbereiche, also Attack, Decay, Sustain und Release lassen sich getrennt bestimmen. Die Einstellzeiten reichen von zwei Millisekunden bis acht Sekunden bei Attack, von sechs Millisekunden bis 24 Sekunden jeweils bei Decay und Release.

Auch die Filtermöglichkeiten wurden alle verwirklicht: Tiefpaß, Bandpaß, Hochpaß sowie fünf Mischformen. Filterfrequenz und Gütefaktor sind regelbar. Die Werte werden in der vierten Zeile links am Bildschirm dargestellt.

Komfortabel gerieten auch die Ringmodulations- und Syncmöglichkeiten. Acht verschiedene Arten der Beeinflussung der DCOs untereinander stehen zur Auswahl. Und hiermit erschließt sich Synthimat 64 das weite Feld von Ufo- und Flipperklängen, Gongs und Meeresbrandung.

Es dauert gewiß einige Zeit, bis man sich durch die Vielzahl der Möglichkeiten, die auch verwöhnten Ansprüchen genügen, hindurchprobiert und gelernt hat, bestimmte Sounds mit dem Commodore 64 zu programmieren. Doch dies geht Musikprofis mit weit teureren Anlagen noch viel schlimmer. Durchhalten lautet die Devise!

An Möglichkeiten mangelt es Synthimat 64 sicher nicht. Manch ordentlicher Synthi mit weit höherem Preis bietet weit weniger! Das Programm kostet ganze 99 Mark.

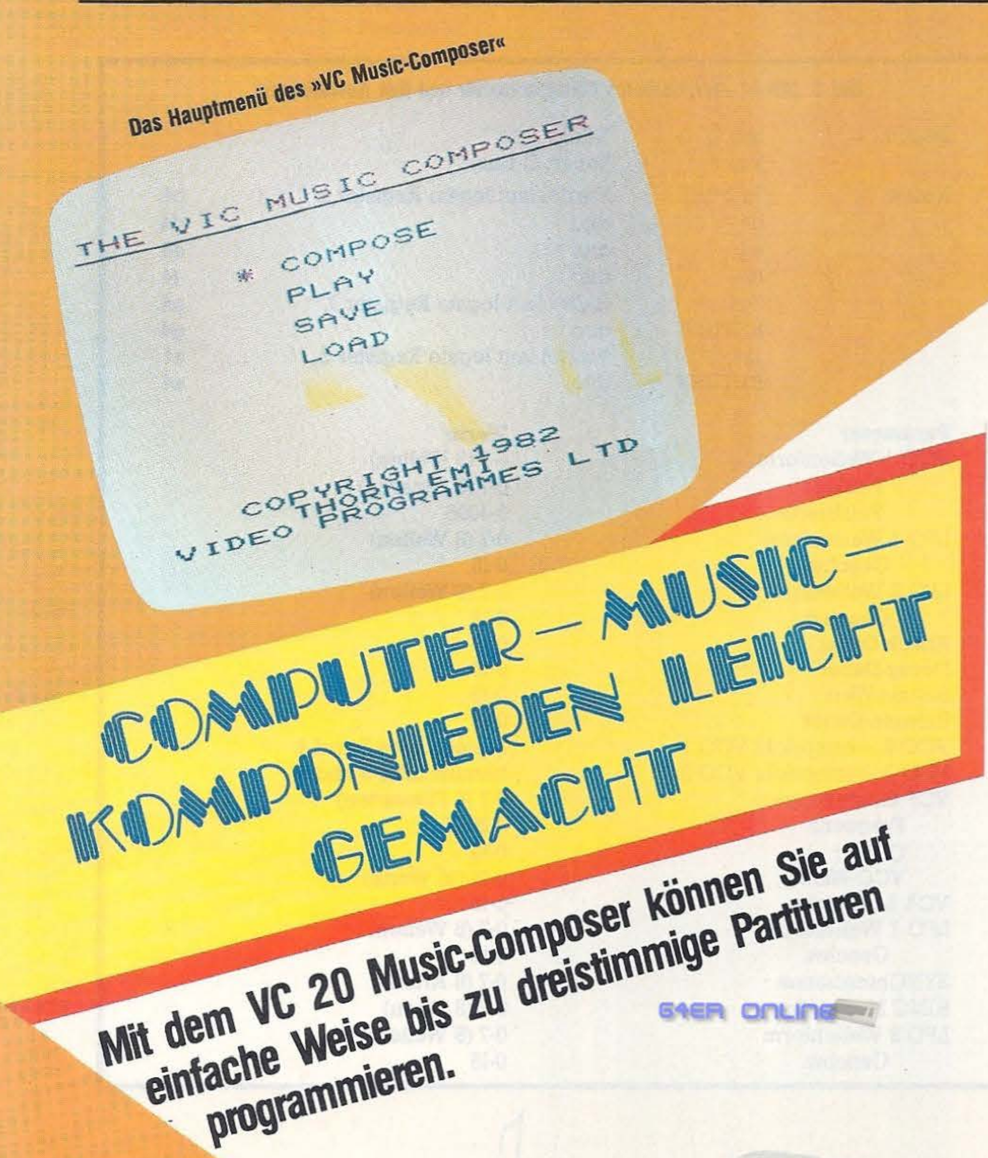
(Richard Aicher/aa)

Bild 2. Die 44 veränderbaren Klangparameter und Ihre Abstufungen

Beginn	tact 4/4	4/4-Takt	
	key c	Tonart C-Dur	
Noten:	1c4;*;8;1	Viertel laut legato Register 1.	c4
	d4	dito	d4
	e4	dito	e4
	f4	dito	f4
	2g4	Halbe laut legato Register 1.	g4
	RETURN	dito	g4
	1a4;*;*;2	Viertel laut legato Register 2,	a4
	RETURN	dito	a4

Parameter	**Werte**
VCO 1 Wellenform	0-7 (8 Wellen)
Fußlage	0-7 (8 Oktaven)
Pulsbreite	0-4095
LFO 1 Wellenform	0-7 (8 Wellen)
Geschw.	0-15
LFO 2 Wellenform	0-7 (8 Wellen)
Geschw.	0-15
Attack-Dauer	0-15
Decay-Dauer	0-15
Sustain-Wert	0-15
Release-Dauer	0-15
VCO 2 —entspricht VCO 1	benutzt LFos 3 und 4
VCO 3 —entspricht VCO 2	benutzt LFos 5 und 6
VCF Einstellungen	0-7 (8 Filterarten)
Frequenz	0-2047
Q-Wert	0-15
VCO-Wahl	0-15 (16 Werte)
VCA Lautstärke	0-15
LFO 7 Wellenform	0-7 (8 Wellen)
Geschw.	0-15
SYNChronisation	0-7 (8 Arten)
RING Modulation	0-7 (8 Arten)
LFO 8 Wellenform	0-7 (8 Wellen)
Geschw.	0-15





Das Hauptmenü des »VC Music-Composer«

COMPUTER – MUSIC – KOMPONIEREN LEICHT GEMACHT

Mit dem VC 20 Music-Composer können Sie auf einfache Weise bis zu dreistimmige Partituren programmieren.

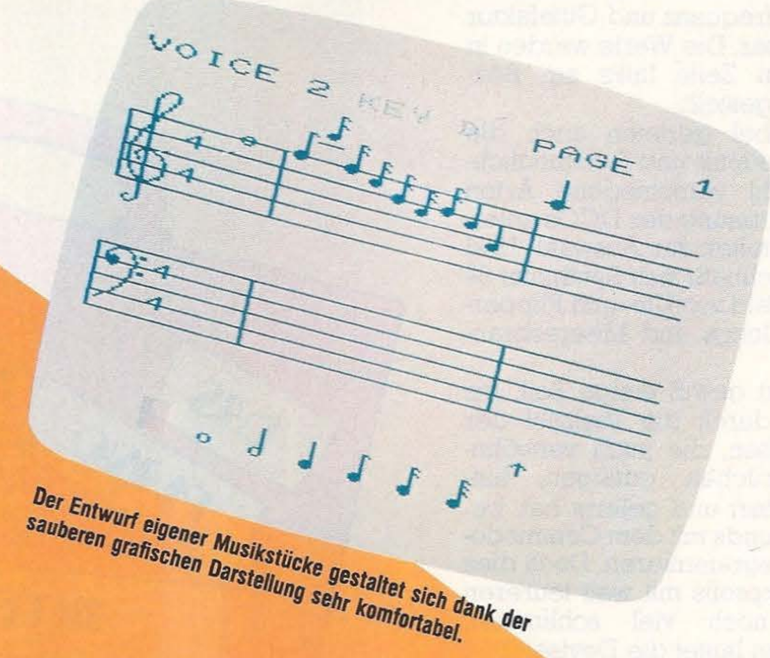
Das Programm arbeitet nicht etwa mit Zahlen oder Buchstaben, sondern benutzt die gewohnte Standardnotenschrift.

Im Compose-Modus fordert Sie der Computer auf, zwischen drei Stimmen zu wählen:

Voice 1 (Alt)
Voice 2 (Tenor)
Voice 3 (Sopran)

Haben Sie sich nun für eine der drei Stimmen entschieden, wird nach der gewünschten Tonart gefragt. Wie jeder Musiker weiß, gehört zu einer vernünftigen Komposition auch ein Takt. Bei der Takteingabe werden allerdings auch äußerst seltsame Takte akzeptiert (zum Beispiel 3/7 Takt, 1/9 Takt etc.).

Nach der Taktwahl erscheint auf dem Bildschirm die Komponier- beziehungsweise Bearbeitungsseite (EDIT-PAGE). Mit den Cursor-Tasten können Sie nun die gewünschte Note (ganze Note bis 1/32 Note) anwählen und diese in eine Notenzeile setzen. Auf Wunsch kann diese Note nun erhöht, erniedrigt oder punktiert werden. Entsprechend lassen sich auch beliebig lange Pausen setzen. Die Taktstriche müssen Sie sich leider selber setzen.



Der Entwurf eigener Musikstücke gestaltet sich dank der sauberen grafischen Darstellung sehr komfortabel.

Wie laut eine Passage gespielt werden soll, wird jeweils in italienischen Begriffen angegeben. Es stehen sechs Lautstärken, von pianissimo (= sehr leise) bis fortissimo (= sehr laut) zur Verfügung.

Die Editiermöglichkeiten des Programms sind gut. Noten lassen sich bequem einfügen oder löschen.

Da sich alle drei Stimmen unabhängig voneinander programmieren lassen, können auch polyphone (= mehrstimmige) Partituren gespielt werden. Dies geschieht im PLAY-Modus. Nach der Frage DISPLAY WHICH VOICE? kann die Notation der gewünschten Stimme parallel zur Melodie verfolgt werden. Das Tempo läßt sich stufenweise von 1 (= sehr schnell) bis 9 (= sehr langsam) anwählen.

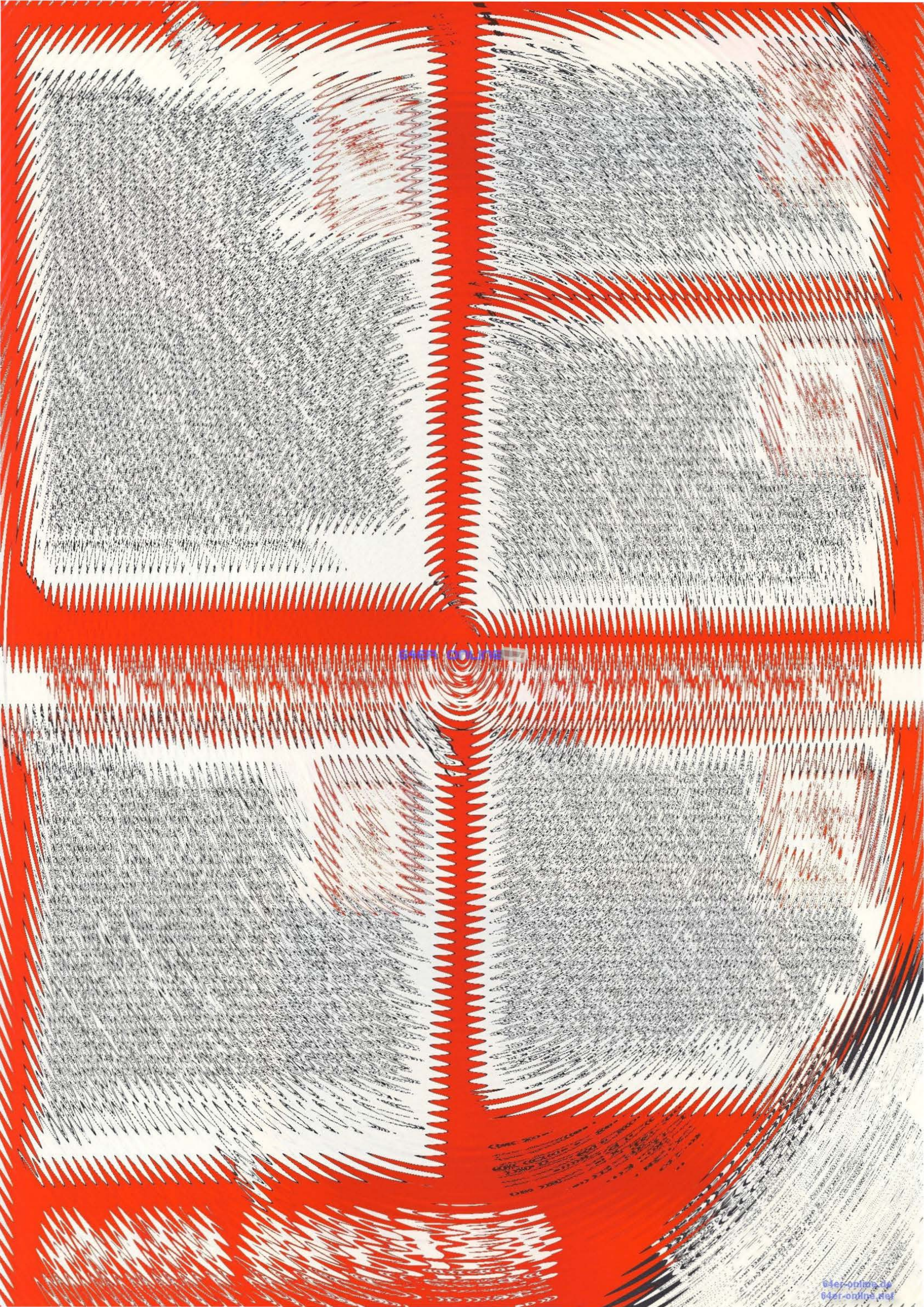
Damit Sie nicht jedesmal das 4. Brandenburgische Konzert von Johann Sebastian Bach neu eingeben müssen, können Sie jedes Stück wahlweise auf Kassette oder Diskette abspeichern und später erneut laden.

Fazit:

Sowohl für den musikalischen Anfänger als auch für den Fortgeschrittenen bietet der VC 20 Music-Composer zahlreiche Möglichkeiten, musikalische Ideen zu verwirklichen.

‚Computer Music' von Thorn Emi Video ist als Steckmodul für den VC 20 erhältlich.

(C.Q.Spitzner/B.Carli)



Brandaktuell aus USA

musik neuigkeiten

Ein weiteres Musik-Keyboard für den Commodore 64

Auch in den USA präsentierte man ein Keyboard für den Commodore 64. Das Colortone Synthesizer Keyboard. Drei Oktaven besitzt das von Waveform (Musicalc!) entwickelte Keyboard und ein futuristisch anmutendes Design, das auch jedem wirklichen Synthesizer alle Ehre machen würde. Mit dem Keyboard können Melodien per Klaviatur in die Musicalc I Software eingespielt werden, was bisher nicht möglich war. Die Songs lassen sich dann wie bekannt mit Musicalc bearbeiten oder in Notenschrift ausdrucken. Somit wäre das System Commodore 64, Floppy 1541, Monitor, Colortone-Keyboard und Musicalc-Software ein komplettes Musiksystem, das auch Keyboarder spielen können, ohne sich die Finger auf einer QWERTY—Tastatur zu verrenken. Der Preis des Colortone-Keyboards wird sich in den USA zwischen 200 Dollar und 300 Dollar bewegen.

Ein weiteres dreistimmiges, vier Oktaven umfassendes Musik-Keyboard, mit dem man die drei Soundgeneratoren des SID-Chips kontrollieren kann, ist seit kurzem in England erhältlich. Es besitzt zwei Slide-Regler, deren Funktion man per Software selbst definieren kann. Hiermit lassen sich zum Beispiel Vibrato-, Pitchbend- oder ähnliche Effekte bequem regeln. Zu diesem Keyboard wird passende Musiksoftware mit Sequenzer und Realtime-Spielmöglichkeiten geliefert. Man arbeitet momentan an einer Softwareerweiterung, mit der dann zirka 0,8 Sekunden Naturklang aufgenommen, gespeichert und per Keyboard über vier Oktaven gespielt werden. Das Microsound 64-Keyboard ist bisher leider nur in England erhältlich. Ich werde sobald als möglich darüber berichten.

MIDI-Interface- und Software auch in einer C 64-Version erhältlich

Roland vertreibt ein intelligentes MIDI-Interface mit eigenem Mikroprozessor, das mittels Interface-Karte an Apple II und IBM PCs angeschlossen werden kann. In Kürze erscheint nun auch eine Version für den Commodore 64. Das Interface verfügt über diverse Synchronisationsmöglichkeiten (Tape Sync, Sync von Roland-Drumy-Units) und erspart durch seine eigene Intelligenz dem externen Computer viel Arbeit. Die C 64-MIDI-Software existiert momentan ebenfalls nur in einer Apple II- und IBM-Version, man schreibt aber gerade fleißig auf den C 64 um. Diese Roland MIDI-Recorder-Software für den Commodore 64 wird dann, genau wie die bisherigen Versionen, acht Tracks zur Verfügung stellen, die 16 Kanäle ansprechen können. Timing-Fehler lassen sich nach der Aufnahme automatisch korrigieren. Vor jeder Aufnahme kann man die Anzahl der aufzunehmenden Takte und der Vorzähl-Takte bis zum Aufnahmebeginn bestimmen.

Microsound 64 — Keyboard mit Musiksoftware für den Commodore 64

Weitere Features: Metronom Temporegelung, Pitchbend. Das Interface kostet zirka 600 Mark, die zugehörige Software mit Interfaceadapter voraussichtlich zirka 300 Mark.

IMA, die Internationale MIDI Association bietet Mitgliedschaft an

Alle an der Entwicklung des MIDI-Systems Interessierte, können in der International Midi Association (IMA) Mitglied werden. Die IMA ist eine nichtkommerzielle Einrichtung. Sie verfügt über sämtliche Informationen zum aktuellen Stand des MIDI-Geschehens. Die Gesellschaft versucht, ein weltweites Forum des Gedankenaustausches zu sein. Die Mitglieder unterteilt man in drei Gruppen: Hersteller, Händler und Anwender. Für jede Kategorie werden spezifisch zugeschnittene Informationen angeboten.

Der Mitgliedsbeitrag beläuft sich für Anwender auf jährlich 40 Dollar, zuzüglich 5 Dollar Postgebühren. Dafür erhält man das »MIDI Specification Manual« kostenlos. Diese Fundgrube für MIDI-Technik-Freaks kann man auch als Nichtmitglied, zum Preis von 10 Dollar zuzüglich 5 Dollar Postgebühren, beziehen.

Daneben existieren noch eine Reihe anderer interessanter Angebote, die Mitglieder kostenlos oder zumindest ermäßigt erhalten. Zum Beispiel, das monatlich erscheinende »IMA Bulletin« vollgepackt mit den allerneuesten Informationen zum MIDI-Standard, Produktinformationen, Kontaktadressen anderer Mitglieder, Seminarpläne und Termine, der 36 mal jährlich erscheinende »IMA Update Service« und jährlich herausgegebene »IMA Sourcer« mit Nachrichten über MIDI Equipment. Detailliertere Informationen bietet die »IMA Membership Information Brouchure«. Diese erhält man über: IMA — The International MIDI Association, 8426 Vine Valley Drive, Sun Valley, CA 91352

(Richard Aicher/aa)



Bild 1. Das Roland MIDI-Interface mit Interfaceadapter

Glanz und Elend MIDI eines Interfaces

Für die meisten Profi-Keyboarder ist der Computer mittlerweile zum unentbehrlichen Begleiter auf den Bühnen der Welt geworden. 1983 haben sich fast alle Synthesizerhersteller auf ein einheitliches Keyboard-Bus-System, das Midi, geeinigt. Seither ist es auch ohne Lötkolben möglich, mehrere Keyboards, Rhythmusmaschinen und Effektgeräte, Takt für Takt synchron, durch einen Song zu jagen. Und wenn man will, nimmt einem der Computer sogar das Spielen ab. Doch war die Einführung dieses Systems auch mit Schwierigkeiten verbunden.

Noch vor einem Jahrzehnt war ein Synthesizer ein riesiges Klangmonster, vollgestopft mit Analog-Elektronik. Dank moderner Mikroprozessortechnik hat sich dies jedoch innerhalb weniger Jahre grundlegend gewandelt. Synthesizer von heute sind klein und handlich, kaum noch größer als die Tastatur und voll polyphon, also vielstimmig spielbar. Alle Klangparameter lassen sich heute problemlos abspeichern und stehen nach einem Druck auf ein Knöpfchen sofort und jederzeit wieder zum Spiel bereit. Zeit ist kostbar, vor allem auf der Bühne.

Die Arrangements moderner Popsongs konnten dank moderner Studiotechnik unheimlich verfeinert werden. Mit einer 24-Spur-Maschine, einem guten Techniker und viel Zeit, läßt sich heute aus einer Fünf-Mann-Combo ein vielstimmiges Orchester zaubern. Die einzelnen Stimmen spielt man hintereinander auf das Bandgerät.

Doch wie realisiert man solche Meisterwerke moderner Studiotechnik auf der Bühne, live, möglichst ohne Playback? Hier sind wir an dem Punkt, der Keyboarder, die etwas auf sich halten wollen, wohl oder übel zu Computerfreaks werden läßt — denn auch ein Keyboarder hat nur zwei Hände. Dank Midi genügen ihm diese nun auch, die Stimmen, die er nicht mehr selbst zu spielen schafft, übernimmt der Computer. Wenn ein Konzert abläuft, hat der Midi-Keyboarder die Hauptaufgabe schon zu Hause erledigt, nämlich das Einprogrammieren der Songs. Theoretisch könnte der Computer den Keyboarder natürlich voll ersetzen. Doch das wäre unfair, wo bliebe dann noch die Show?



Bild 1. Ein Midi-System live im Einsatz, Manfred Rürup von der Inga Rumpf Band

Midi hilft nicht nur, komplexe Studiomusik live auf der Bühne zu realisieren. Das Inventar eines modernen Tonstudios geht heutzutage in die Millionen. Entsprechend hoch sind auch die Mieten. Midi hilft auch, teure Studiozeit zu sparen. Zumindest Keyboards und Elektronikdrums programmieren viele weniger betuchte Musikgruppen heute per Computer und Midi bereits zu Hause fertig ein. Im Studio spielt dann der Computer fehlerfrei sämtliche Stimmen gleichzeitig auf das Band. Mühseliges und zeitraubendes Spur-für-Spur-Aufnehmen wird so hinfällig. Midi ist also quasi notwendig geworden. Immer komplizierter werdende Technik erfordert meist noch mehr Technik, um sie wieder in Griff zu bekommen.

So, genug des Lobs. Selbst Computermusiker, habe ich mich wieder einmal in Begeisterung geschrieben. Es wäre mittlerweile allerdings sehr wichtig, die anfängliche Begeisterung von Musikern, Herstellern und Presse, über das neue Wunderkind, das alles möglich zu machen versprach, gegen eine kritischere Betrachtungsweise auszuwechseln; Midi, nach einem Jahr auf dem Markt, als das zu betrachten, was es eigentlich wirklich ist, — lediglich ein genormtes Bus-System. Genau wie RS232 oder Centronics, oder S-100 oder wie sie alle heißen. Ein Standard wie viele andere, behaftet mit genau den gleichen Unzulänglichkeiten und Vereinfachungen. In der Entwicklung behindert durch Fehlinterpretationen und dem Konkurrenzverhalten vieler Industriefirmen. Das ist das Fazit nach einem Jahr Midi.

Sehen wir uns an, wie sich alles entwickelte. ►

Die Idee, ein genormtes Bussystem für Synthesizer-Keyboards zu entwickeln, kam erstmals im Juni 1981 auf der berühmten NAMM (National Association of Music Merchants) Show in Anaheim, USA auf. Diese größte Musikmesse der Welt findet jährlich statt, — hier werden meist sämtliche Neuentwicklungen der Musikindustrie erstmals vorgestellt. Die Präsidenten der bekannten Synthesizerfirmen Sequential

Wie sich alles entwickelte

Circuits, Oberheim Electronics und Roland (Dave Smith, Tom Oberheim, Kahehashi) unterhielten sich erstmals über die Möglichkeiten eines Standard-Digital-Interface für ihre Keyboards. Die nächsten Monate arbeitete man an ersten Vorschlägen. Ein weiteres Gespräch im Oktober 1981; der Kreis der Interessenten war mittlerweile gewachsen — Yamaha, Korg und Kawai, drei japanische Hersteller, hatten sich mittlerweile dazugesellt. Im Oktober 1981 stellte dann Sequential Circuits (SCI) auf der AES (Audio Engineering Society) in New York den ersten Versuch eines solchen Interfaces vor. Es hieß USI (Universal Synthesizer Interfaces) und war ein schnelles serielles Interface. Der Effekt: wenig Interesse von seiten anderer Hersteller. Doch der Stein war damit ins Rollen gekommen. Bei der NAMM-Show im Januar 1982 zählten bereits 15 Synthesizer-Hersteller zum Kreis der Interface-Interessenten: Oberheim, E-mu, Yamaha, Korg, Roland, Kawai, Moog, Fairlight, GDS, CBS-Rhodes, Music Technologie Inc., Octave Plateau, Passport Design, Alpha Syntauri und noch einige mehr. Für Keyboarder alles bekannte und wohlklingende Namen. Viele Streitpunkte tauchten auf, man fand keine Einigung, SCI und die Firmen, Yamaha, Roland, Korg, Kawai waren die einzigen, die nach diesem Treffen zunächst weitermachten. Es war mittlerweile klar, daß es kein optimales Interface geben würde, jedes müßte ein Kompromiß sein. Ein Kompromiß zwischen dem Wunsch jedes Herstellers nach wie vor Keyboards zu entwickeln, die sich möglichst von allen anderen in den meisten Features unterscheiden und andererseits dem Wunsch, alle Instrumente unter einen Hut zu bringen. So verwundert es nicht, daß trotz vieler Bemühungen, bis zum heutigen Tage wirklich 100 prozentig mit Midi nur eines machbar ist: die Ansteuerung eines zweiten Keyboards von einem ersten aus. Das heißt: drückt man auf eine Taste von Keyboard 1, klingt der Ton gleichzeitig von beiden Keyboards. Hier treten nie Schwierigkeiten auf, unabhängig davon, welchen Markennamen die gekoppelten Keyboards tragen.

Im folgenden Jahr machten nun die Japaner einige Vorschläge zur Abänderung des USI-Interfaces und Roland erfand den Namen Midi (Musical Instrument Digital Interface).

Im Januar 1983, wieder einmal NAMM, war es dann soweit. SCI stellte das erste Midi-Keyboard vor, den Prophet 600. Auch die Roland-Leute waren fleißig gewesen und hatten ebenfalls ein Midi-Keyboard entwickelt, den JP-6. Und, welch ein Wunder, obwohl die Instrumente unabhängig voneinander entwickelt wurden, verstanden sie sich. Keyboarddaten ließen sich austauschen. Doch das war auch schon alles. Tonhöhe und Anschlagsdauer wurden zwar korrekt übermittelt, SCI übermittelte jedoch zusätzlich Arpeggiator-Daten, Roland nicht.

Nach anfänglicher Begeisterung regte sich bald Mißtrauen unter den Musikern

Erst im Nachhinein wurde man sich leider klar, daß es gewisse Richtlinien geben müsse, welche Daten übertragen werden sollen und wie. So entstand die erste Midi-Spezifikation 1.0. Leider waren zu diesem Zeitpunkt schon Keyboards auf dem Markt, die ihr noch nicht entsprachen. Überdies hatten sich unter den Musikern mittlerweile die unglaublichsten Gerüchte aufgebauscht, was Midi alles könne: nämlich alles. Sie stellten jedoch bald fest, daß dies weit gefehlt war. Die Enttäuschung war groß, die Unsicherheit wuchs. Musiker ahnen eben nicht, welche Schwierigkeiten sich hinter einer so einfach aussehenden Entwicklung verbergen.

Mittlerweile jedoch hatte die anfängliche Begeisterung von seiten der Musiker viele Hersteller, die zunächst der Spezifikation skeptisch gegenüberstanden, in Zugzwang gebracht. Sie befürchteten, ihre Keyboards ohne Midi-Anschluß nicht mehr los zu werden. Die Folge: Mittlerweile sind bis auf vereinzelte Ausnahmen alle neueren Keyboards mit Midi ausgerüstet.

Die Geister waren hiermit gerufen. Es bleibt nur noch, das Beste aus dem Anfang zu machen. Alle Hersteller haben sich mittlerweile geeinigt, bis März dieses Jahres die gültige Midi-Spezifikation 1.0 in allen Punkten zu befolgen. Die Hoffnung bleibt.

Die technischen Details der Midi-Spezifikation 1.0

Wenden wir uns nach diesem historischen Abriß den technischen Details der Midi-Spezifikation 1.0 zu.

Um die Verkabelung der Instrumente einfach zu halten, hat man sich für eine serielle Datenübertragung entschlossen. Diese Tatsache stößt nach wie vor auf viel Widerspruch. Viele halten die Datenübertragungsgeschwindigkeit für zu niedrig. Sie würden lieber mit paralleler Datenübertragung arbeiten. Bei sehr komplexen Systemen, in denen viele Instrumente gleichzeitig mit sehr vielen Daten versorgt werden sollen, können Verzögerungen aufgrund der langsamen seriellen Datenübertragung Delays auftreten, das heißt, die Toninformationen erscheinen am Ende der Leitung mit Verzögerung. Der Streit ist groß, ob dieser Effekt nun hörbar ist oder nicht. Einige wollen Delays störend bemerkt haben, andere entgegnen, daß dieselben Zeitunterschiede für einen Hörer auftreten, der lediglich einige Meter vom Keyboard entfernt steht. Unser Schall ist ja auch nicht der schnellste. Und wer hat hier schon störende Delays bemerkt? Sei es wie es sei. Immerhin arbeiten Midi-Interfaces mit fast der doppelten Geschwindigkeit des in der Computerindustrie weit verbreiteten RS232 Busses, nämlich mit ►

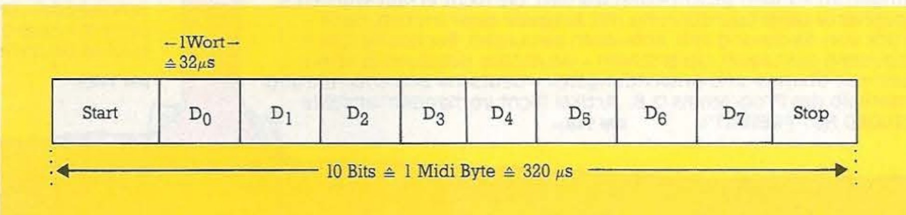
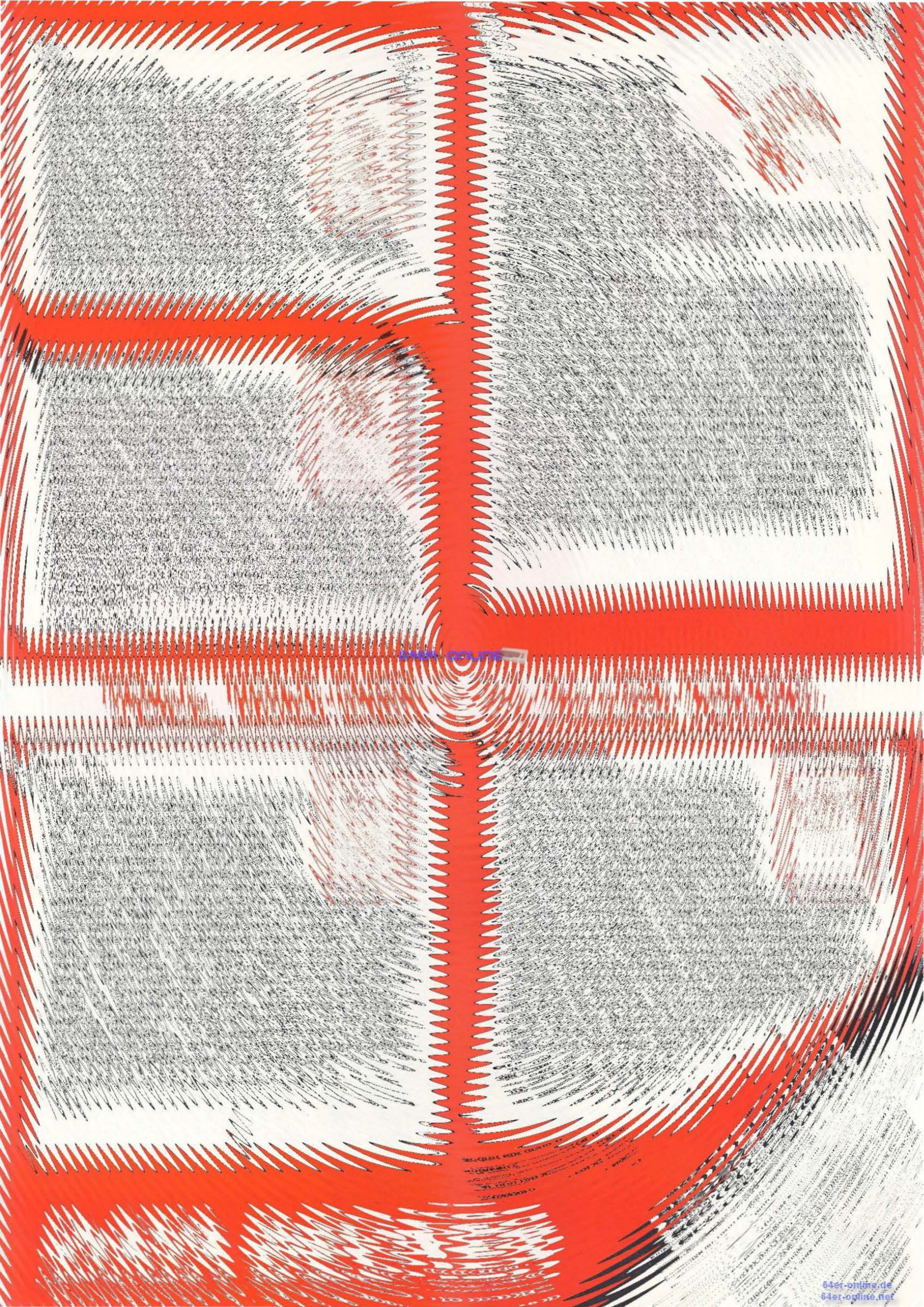


Bild 2. Jedes Midi-Wort besteht aus zehn Bits



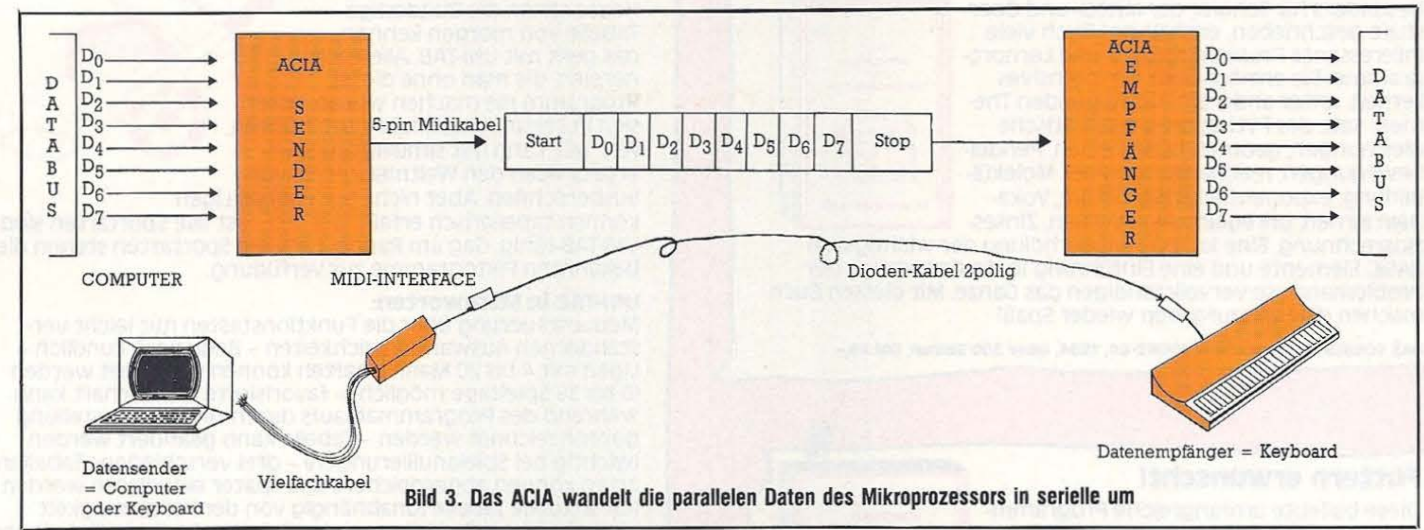


Bild 3. Das ACIA wandelt die parallelen Daten des Mikroprozessors in serielle um

31,25 KBaud. Diese Frequenz wählte man, da sie hardwaremäßig leicht durch Teilung eines 1-MHz-Taktes durch 32 zu erhalten ist. Jedes Daten-Byte besteht aus einem Start-Bit, acht Daten-Bits und einem Stop-Bits (siehe Bild 2).

In jedem mit Midi ausgerüsteten Instrument sowie in allen Midi-Interfaces befindet sich nun eine Baugruppe, die parallele Daten des internen Keyboardprozessors in serielle wandelt und auf die Reise in das Midi-Verbindungskabel schickt, beziehungsweise von einem externen Computer ankommende serielle Daten in parallele rückwandelt. Diese Einheit heißt ACIA (Asynchronous Communications Interface Adapter Bild 3).

Von außen unterscheidet sich ein Midi-Keyboard nur durch drei Buchsen — Midi-Input, Midi-Output und Midi-Durchgangsbuchse (Midi Thru) — von einem anderen. Laut Spezifikation dürfen lediglich 5-Pin/180-Grad-DIN- beziehungsweise XLR-Buchsen Verwendung finden. Verbindungskabel zwischen Midi-Systemen dürfen nicht länger als 15 m sein, man muß eine zweipolige Leitung, abgeschirmt und verdrillt, verwenden.

Will man Keyboards nicht bloß verbinden, sondern von einem Computer ansteuern, benötigt man ein externes Midi-Interface, das die parallelen Daten des Computers in serielle wandelt, die unser Keyboard versteht. Den prinzipiellen Aufbau eines Midi-Interfaces erkennen wir in Bild 4.

Solch externe Interfaces gibt es mittlerweile in den verschiedensten Modifikationen. Viele passen nur an einen Rechner, einige mittels zusätzlicher Adapter-Karten an mehrere. Für den Commodore 64 existieren momentan die meisten Midi-Interfaces und, was noch wichtiger ist, sehr viel Software. Auch im Midi-Sektor scheint der C 64-Konkurrenten mittlerweile weit abzuschlagen.

Bild 4. So ist prinzipiell jedes Midi-Interface aufgebaut

Welche Daten sollen übertragen werden? Diese Frage ließ Köpfe rauchen

Das Problem des Midi-Standards lag bisher jedoch weniger in der Normung der Hardware. Ungleich schwerer fiel die Entscheidung, welche Daten nun eigentlich übertragen werden können beziehungsweise müssen, um das System für den Musiker interessant zu machen.

Optimal wäre, alle Features eines Instruments per Midi auf ein anderes Instrument übertragen zu können.

Dieser Wunsch scheitert jedoch sofort an den unterschiedlichen technischen Möglichkeiten diverser Geräte. Midi kann lediglich auf dem niedrigsten gemeinsamen Level aller angeschlossenen Instrumente wirken. So lassen sich zwar auf einen vierstimmigen Synthesizer Daten für acht Stimmen, also eine achtstimmige Komposition, übertragen, gleichzeitig spielen wird er aber nur vier hiervon. Ebenso werden einen Synthi, der keine Anschlagsdynamik aufweist, diesbezügliche Daten kalt lassen. Die erste Lektion lautet folglich: Midi erweitert niemals die technischen Möglichkeiten angeschlossener Instrumente!

Zweitens: Jedes Instrument interessieren nur ganz bestimmte Informationen; Keyboards ganz andere als zum Beispiel eine elektronische Rhythmuseinheit oder einen Sequenzer. Um Synthesizer sinnvoll zu koppeln, müssen mindestens die Keyboardinformationen, Tonhöhe, Gate on time und die Anschlagsgeschwindigkeit beziehungsweise -dynamik codiert übertragen werden.

Polyphone Sequenzer können mit denselben Daten arbeiten, nicht aber monophone Sequenzer. Letztere registrieren nur einzelne Melo- ►

diestimmen. Rhythmusmaschinen wiederum sind überhaupt nicht an Ton-, sondern nur an Synchronisationsinformationen interessiert.

Noch schwieriger vorhersehbar als diese Punkte ist die Frage, in welcher Konfiguration von Geräten das Midi-Interface dann jeweils tatsächlich vom Benutzer eingesetzt werden wird. Will man zum Beispiel mehrere verschiedene Synthis ansteuern und auf jedem eine andere Stimme einer mehrstimmigen Komposition mit anderem Klang ausgeben, so erfordert dies sicher eine ganz andere Zuordnung, als wenn alle Instrumente dieselbe Stimme spielen sollen. Deshalb führte man drei unterschiedliche Zuordnungsmodi ein, den OMNI-, POLY- und MONOmodus. Da die Verkabelung der einzelnen Instrumente möglichst einfach sein soll, müssen softwaremäßig Kanäle geschaffen werden, um die Instrumente mit den unterschiedlichen Stimmen der Komposition zu versorgen. Jedes Instrument muß für es bestimmte Daten gezielt erkennen. So führte man neben den drei Modi noch 16 Kanäle (Channels) ein. Mit geeigneter Software ließen sich deshalb auch maximal 16 Instrumente gleichzeitig und polyphon ansprechen — angesteuert von einem Computer.

Was für Möglichkeiten bieten die einzelnen Modi? Im OMNImodus spielen alle Instrumente, die am Bus hängen, parallel und polyphon. Die angeschlossenen Instrumente empfangen sämtliche über den Bus geleitete Daten, unabhängig vom jeweiligen Kanal, auf dem diese übermittelt werden. Sollen die gekoppelten Synthis jedoch verschiedene Stimmen spielen, wechselt man in den POLYmodus. Hier lassen sich die einzelnen Instrumente unterschiedlich adressieren. Kanal 1 spricht dann zum Beispiel nur Synthi 1 an. Man könnte auf diesem Kanal eine Baßstimme programmieren, über Kanal 2 einen zweiten Synthesizer ansprechen und diesen Begleitakkorde spielen lassen. Eine Melodiestimme über Kanal 3 auf Synthi 3 gelegt und zu guter Letzt eine Rhythmusmaschine synchronisiert, über Kanal 4 — schon hat man das Orchester fertig. Auf jedem Kanal können theoretisch unbegrenzt viele Stimmen gleichzeitig übermittelt werden; das angeschlossene Keyboard spielt natürlich nur so viele, wie es Stimmen besitzt. Bei modernen Keyboards sind das mittlerweile bis zu 16, wie der Yamaha DX-7 zeigt. Hätte man 16 DX-7, könnte man eine 16 x

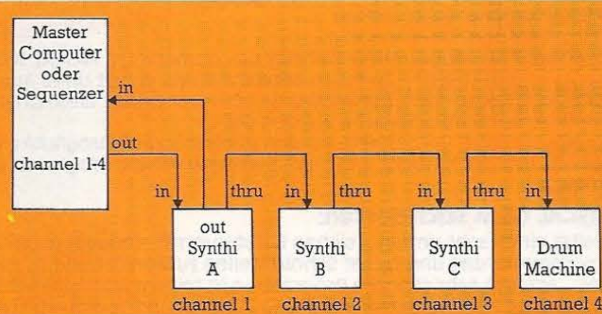


Bild 5. Sternförmige und kettenförmige Verkabelung von Computer, Midi-Interface, Synthesizern und Rhythmusmaschinen

16, also 256stimmige Komposition mit Top-Sound, von einem Commodore 64 gesteuert, über diese Maschinerie ausgeben! Da die jüngsten Tendenzen der Synthesizerindustrie wieder in Richtung Modultechnik gehen, steht dies auch nicht mehr in allzu weiter Ferne. So erscheinen demnächst von den Firmen Roland und Yamaha 19-Zoll-Synthiracks, in denen jeweils acht komplette Synthesizer untergebracht sind, per Midi und Computer steuerbar. Dann braucht man nur noch ein Klaviatur-Modul und los geht's.

Im dritten und letzten Modus, dem MONOmodus wird schließlich je Kanal nur eine einzige Stimme übertragen. Jedes angeschlossene Instrument empfängt so viele Kanäle, wie es Stimmen besitzt, beginnend bei dem Kanal, mit dem es adressiert wurde. Verfügt es über keinen Adreßschalter, wird es automatisch mit Kanal 1 adressiert. Letzteres gilt übrigens für alle Modi. Besitzt man zum Beispiel den Six-Track von Sequential Circuits, einen Synthesizer, der sechs verschieden klingende Stimmen gleichzeitig spielen kann, lassen sich diese Stimmen im MONOmodus mit unterschiedlichen Melodien belegen. So spielt dann ein Synthi gleichzeitig stampfendes Baßfundament, Bläserbegleitung, Fuzzsolo und vielleicht noch drei Percussionstimmen — was will man mehr.

Prinzipiell sind zwei verschiedene Arten der Verkabelung angeschlossener Geräte möglich, sternförmig und kettenförmig. Wir sehen dies in Bild 5.

Alle Midi-Features funktionieren meist nur im Idealfall

Midi bietet also ungeheure Möglichkeiten. Doch das Ganze funktioniert tadellos nur im Idealfall.

Vor dem Kauf jeglichen Midi-Equipments sollte man sich zunächst immer möglichst genau informieren, ob das Gerät auch in Zusammenhang mit den anderen, die man schon besitzt, funktionieren wird. Schwierigkeiten treten meistens auf, wenn Equipment verschiedener Hersteller gekoppelt werden soll. Die meisten Möglichkeiten und wenigsten Probleme ergeben sich, wenn man nur Equipment eines Herstellers nutzt. Dies ist prinzipiell nicht anders als in der Computerszene. Andernfalls werden zwar viele Features funktionieren, aber nicht alle.

Was man dann mit dem Computer und seinem Equipment tatsächlich für Möglichkeiten hat, bestimmt die Qualität der verwendeten Software. Midi-Software für den Commodore 64 gibt es mittlerweile in Hülle und Fülle. Tunlichst sollte man sich auch hier vor dem Kauf genau über deren Möglichkeiten informieren. Sich wenn möglich alles praktisch demonstrieren lassen. Hier liegt jedoch meist der Hund begraben. Man versuche einmal, in einem Musikgeschäft bestimmte Midi-Software vorgeführt zu bekommen. Die meisten werden passen. Mangels Kenntnis im Umgang mit dem Computer oder überhaupt in Ermangelung eines solchen. Entsprechend wird man ebenso verzweifelt in Computershops nach Midi-Keyboards Ausschau halten. Hier bleibt nur die Hoffnung, daß sich demnächst einiges ändert.

(Richard Aicher/aa)



Die Index-Sequ

Das Arbeiten mit einer relativen Datei ist gar nicht so komfortabel wie man es eigentlich möchte. Der Zugriff auf Datensätze nur über die Datensatznummer ist nicht besonders benutzerfreundlich und in der Regel auch praxisfremd. Unter Zuhilfenahme einer sequentiellen Datei wird dieser Nachteil jedoch aufgehoben.

In den letzten beiden Ausgaben des 64'er wurden Ihnen die Grundzüge der sequentiellen und der relativen Datei vorgestellt und erläutert. Ich deutete auch schon an, daß erst die Verbindung dieser beiden Dateiformen zu einer befriedigenden Lösung führt. Aber auch hier muß man Abstriche machen. Durch diese Vermischung werden leider nicht nur die Vorteile, sondern auch ein Teil der Nachteile mit übernommen. Ich möchte diese noch einmal kurz anreißen.

Das Wesen der sequentiellen Datei liegt in der aufeinanderfolgenden (eben sequentiellen) Anordnung der Daten. Um mit dieser Datei arbeiten zu können, muß sie vollkommen in den Speicher des Computers geladen werden. Bei großen Datenbeständen kann das eine ganze Zeit lang dauern. Andererseits wird die maximale Größe durch den zu Verfügung stehenden Speicherplatz im Computer selbst bestimmt. Sind jedoch alle Datensätze erst einmal geladen, ist ein Bearbeiten der Daten sehr einfach und auch schnell. Dann können alle Änderungen, Ergänzungen und sonstige Manipulationen im Computer selbst durchgeführt werden. Und für zeitkritische Funktionen wie Sortieren und Suchen kann man die Maschinensprache sehr wirkungsvoll einsetzen.

Die relative Datei hingegen benötigt außer dem eigentlichen Programm nur sehr wenig Speicherplatz, nämlich nur noch denjenigen für die internen Variablen und für einen Datensatz. Der Rest aller Daten befindet sich auf Diskette. Und dort stehen einem fast 170 KByte zur Verfügung. Allerdings ist das Arbeiten mit einer rein relativen Datei nur in speziellen Fällen sinnvoll und praktisch, da bei ihr ein Datensatz nur über die Datensatznummer angesprochen werden kann. Ein weiteres Merkmal ist, daß die Datensatzlänge vorher festgelegt werden muß und eine Floppy unbedingt notwendig ist.

Indizierung als Lösung

Durch die Verbindung einer relativen Datei mit einer sequentiellen wird ein Kompromiß geschlossen. Einerseits wird die Suche nach einem Datensatz, beispielsweise nach einer Adresse, jetzt direkt über den Namen möglich. Andererseits muß gleich nach dem Programmstart die sequentielle Datei geladen und am Ende wieder gespeichert werden. Allerdings fällt sie wesentlich kürzer aus. In ihr braucht lediglich ein Feld der Datensätze enthalten zu sein, nämlich das Indexfeld, im Beispiel die Nachnamen, und nicht, wie bei der puren sequentiellen Form, alle Felder.

Gehen wir einmal von folgender Voraussetzung aus: Auf der Diskette sind bereits eine Anzahl von Adressen gespeichert. Im RAM des Computers befindet sich das Programm und in einem eindimensionalen Feld (im Beispiel die Variable IN$) alle bisher eingegebenen Nachnamen. Indem wir diese Variable mittels einer FOR-NEXT-Schleife durchsuchen, können wir auch einen gespeicherten Namen finden. Allerdings kommen wir damit noch nicht an seine Adresse heran, da diese ja in diesem Moment nicht im Computer sondern auf Diskette gespeichert steht. Mit einem kleinen Kniff schaffen wir aber auch diese Hürde. Wir speichern bei der Eingabe einer neuen Adresse einfach die Datensatznummer vor dem Nachnamen in die Variable IN$. Und zwar reservieren wir vor jedem Nachnamen vier Stellen für die zugehörige Datensatznummer. Ab der 5. Stelle beginnt also der jeweilige Nachname. Wollen wir nun einen Namen suchen, so müssen wir mit der MID$-Funktion die Feldvariable IN$ ab dem fünften Buchstaben durchkämmen. Sobald der gesuchte Name gefunden wurde, trennen wir die ersten vier Zeichen ab und erhalten damit die zugehörige Datensatznummer. Mit dieser Nummer können wir jetzt direkt auf die relative Datei auf der Diskette zugreifen und holen uns die komplette Adresse.

... und so wird's gemacht

Anhand eines Beispiels soll das noch einmal gezeigt werden:
Inhalt der ersten Elemente von IN$:
IN$(1) =»1... MEIER«

...uentielle Datei

IN$(2) =»2... MUELLER«
IN$(3) =»3... ANDERMANN«
IN$(4) =»*«

Gesuchter Name=Mueller, er steht in der Variablen N$ (Zeile 3060)
3060 INPUT"NACHNAME";N$
3070 N=LEN(N$) Länge des eingegebenen Namens

Die folgenden Programmzeilen suchen den Namen.
3090 FOR I=1 TO DATEIENDE
3100 IF IN$(I) = "*" THEN 3120
3110 IF MID$(IN$(I),5,N)=N$ THEN gefunden
3120 NEXT I

Zur Zeile 3100 kommen wir später noch. Angenommen, wir haben nicht den ganzen Namen eingegeben, sondern lediglich die ersten drei Buchstaben MUE. Dann werden in der Variable IN$ nur die 5.,6. und 7. Zeichen, das entspricht den ersten drei Buchstaben eines jeden Namens, verglichen. Dafür sorgt die Zeile 3070 und in 3110 das »,N« als letzter Parameter der MID$-Funktion. Somit könnte mit dieser Eingabe auch der Name MUECKE gefunden werden. Im Extremfall, wenn nichts eingegeben, also RETURN gedrückt wird, findet diese Funktion jeden Namen. Das heißt, man kann mit der Suchfunktion sich nicht nur einzelne Adressen holen, sondern auch die gesamte Datei, oder zum Beispiel alle Adressen, die mit M anfangen. Damit würde der Menüpunkt »G = Anzeigen gesamte Datei« überflüssig. Ich habe ihn nicht entfernt, da hier unabhängig von der sequentiellen Datei alle Datensätze direkt von der Diskette aus der relativen Datei gelesen werden. Man könnte diese Funktion benutzen, um eine eventuell (zum Beispiel durch Stromausfall) teilweise zerstörte sequentielle Datei zu reorganisieren.

Die Zeile 3100 wird dann verständlich, wenn man sich den Programmteil »Neue Datei anlegen« ab 11000 anschaut. In den Zeilen 11320 bis 11350 wird jedes Element der Variablen IN$ mit einem »*« vorbesetzt. Der Stern kennzeichnet einen leeren Datensatz. Damit kommen wir auch gleich zum Programmteil »Löschen Datensatz« ab 4000. Auch dort wird ein zu löschender Datensatz in IN$ mit einem »*« gekennzeichnet und zusätzlich der entsprechende Datensatz auf der Diskette mit Hex FF (= Dezimal 255 = das Zeichen PI). Auch das wollen wir uns anhand eines kleinen Beispiels näher betrachten:

Wir wollen den Herrn Mueller aus unserer Datei entfernen. Nachdem wir seine Adresse mit der Suchfunktion gefunden haben, geben wir ein »L« für Löschen ein. Danach springt das Programm nach 4000, schreibt in die Variable IN$ das »*«, überschreibt den Datensatz auf der Diskette mit CHR$(255) (=PI), löscht die einzelnen Datensatzfelder (NN$, NV$, und so weiter) und springt zurück nach 3350. Die Variable IN$ sieht jetzt so aus:
IN$(1) =»1... MAIER«
IN$(2) =»*«
IN$(3) =»3... ANDERMANN«»
IN$(4) =»*«

Der zweite Datensatz ist nun mit »*« als gelöscht und leer gekennzeichnet. Das bedeutet auch, daß er wieder mit einer neuen Adresse belegt werden kann. Sehen wir uns dazu den Programmteil »Neueingabe Adresse« ab 1800 an.

Nachdem wir eine neue Adresse eingegeben haben (die einzelnen Programmteile, die in 1830 bis 1850 aufgerufen werden, kennen Sie ja bereits aus den letzten beiden 64'er Ausgaben), muß jetzt ein leerer Platz gefunden werden, auf dem wir die neue Adresse abspeichern können. Wichtig sind jetzt die Zeilen 1880 bis 1910. In diesem Abschnitt wird IN$ so lange durchsucht, bis ein mit »*« gekennzeichnetes Element gefunden wird (1910). In unserem Beispiel ist das bereits das zweite Element (IN$(I)=»*«, bei I=2). Somit wird die neue Adresse als zweiter Datensatz auf Diskette gespeichert. In Zeile 1930 und 1940 wird die Variable IN$ auf den neuesten Stand gebracht. Zuerst wird die Datensatznummer (I) in einen String umgewandelt (1920). Dabei wird immer das (unsichtbare positive) Vorzeichen mit berücksichtigt. Da wir dieses nicht brauchen, wird es abgeschnitten (MID$(I$,2) und die Zahl mit Leerstellen auf insgesamt vier Zeichen Länge erweitert. Zeile 1940 verbindet dann die Datensatznummer mit dem Namen.

Vielleicht ein Wort noch zur MID$-Funktion. Normalerweise gehören zu dieser Funktion drei Parameter. So steht es auch kurz erläutert im Commodore-Handbuch. Der letzte Parameter gibt an, wieviele Zeichen ab einer definierten Stelle des Strings genommen werden sollen. Wird dieser Parameter weggelassen, werden ab der definierten Stelle alle restlichen Zeichen des Strings übernommen.

Probleme, die sich ergeben können

Es könnte sein, daß Ihnen die Dateistruktur nicht gefällt. Zum Beispiel möchten Sie die Länge der Datenfelder ändern. Das dürfen Sie nur machen, wenn die Datei neu eingerichtet werden soll. Bei einer bestehenden geht es nicht! Beim Verkürzen gibt es keine Probleme, auch keine beim Verlängern bis zu 88 Zeichen. Zu verändern sind dann die Zeile 30040 sowie die Längenangaben zwischen den Zeilen 5000 und 7100. Bedenken Sie dabei, daß die Variable BL$ mindestens so viele Leerstellen haben muß, wie das längste Datenfeld.

Schwieriger wird es, wenn Sie infolge einer Vergrößerung der Datensatzlänge über 88 Zeichen hinwegkommen. Das Problem hierbei ist der INPUT#-Befehl. Mit ihm kann man nämlich nur maximal 88 Zeichen aus einer Datei lesen. Es gibt zwei Methoden, diese Hürde zu überwinden. Erstens eine Maschinensprache-Routine, die den INPUT#-Befehl erweitert, so daß auch Datensätze mit bis zu 255 Zeichen gelesen werden können (schauen Sie doch einmal in das Floppy-Buch von Data Becker, dort finden Sie eine entsprechende Routine). Die zweite Möglichkeit ist die Verwendung des GET#-Befehls anstelle des INPUT#-Befehls. Dann ist die

Die Index-Sequent

Listing Index-sequentielle Datei

```
100 REM  ******************************
110 REM  * ADRESSENDATEI 64'ER/9      *
120 REM  * INDEX-SEQUENTIELL          *
130 REM  ******************************
140 :
150 :
160 GOSUB 30000:REM  INIT
170 CLOSE 1:OPEN 1,8,2,FR$+",L,"+CHR$(DL
)
180 CLOSE 3:OPEN 3,8,3,FI$+",S,R"
190 GOSUB 10000:REM :DISKFEHLER
200 IF A1<>0 THEN RUN
210 INPUT# 3,IN$:MX$=LEFT$(IN$,15)
220 MX=VAL(MX$)
230 :
240 IN$(0)=IN$
250 PRINT "□"
260 PRINT :PRINT :PRINT
270 PRINT "          INFORMATION"
280 PRINT :PRINT
290 PRINT "  BISHERIGE DATEIGROESSE: ";M
X
300 PRINT :PRINT
310 PRINT "         BITTE WARTEN"
320 I=0
330 I=I+1
340 :INPUT# 3,IN$(I):PRINT ""I;MX;IN$(I)
350 IF ST<>64 THEN 330
360 PRINT :PRINT
370 PRINT "  DRUECKEN SIE EINE TASTE"
380 POKE 198,0:WAIT 198,1
390 REM  ---------------------------
1000 REM  - MENUE                   -
1010 REM  ---------------------------
1020 :
1030 PRINT "□"
1040 PRINT :PRINT
1050 PRINT "        ADRESSENDATEI"
1060 PRINT "  RELATIV UND SEQUENTIELL"
1070 PRINT :PRINT
1080 PRINT "■ X = PROGRAMMENDE"
1090 PRINT
1100 PRINT "   G = ANZEIGEN GESAMTE DATE
I
1110 PRINT
1120 PRINT "   S = SUCHEN"
1130 PRINT
1140 PRINT "   N = NEUE ADRESSEN EINGEBE
N"
1150 PRINT
1160 PRINT "   ! = NEUE DATEI ANLEGEN"
1170 PRINT :PRINT :PRINT :PRINT
1180 PRINT "WAEHLEN SIE ";
1190 POKE 198,0
1200 GET R$:IF R$="" THEN 1200
1210 IF R$="X" THEN CLOSE 1:GOSUB 15000:
CLOSE 15:END
1220 IF R$="G" THEN GOSUB 3500:REM  ANZ.
1230 IF R$="S" THEN GOSUB 2570:REM  SUCH
1240 IF R$="N" THEN GOSUB 1800:REM  NEUE
ING.
1250 IF R$="!" THEN GOSUB 11000:REM  NEU
DATEI
1260 GOTO 390
1800 REM ---------------------------
1810 REM  SCHREIBEN /EINGABE SEQ/REL-
1820 REM ---------------------------
1825 NN$=".":NV$=".":OT$=".":TE$=".":PL$
=".":SR$="."
1830 GOSUB 2000:REM  AUSGABE 1 DATENSATZ
1840 GOSUB 6000:REM  EINGABE
1850 GOSUB 7000:REM  VERKETTEN
1860 REM  BESTIMMUNG SATZNUMMER
1870 LZ$="    ":LZ=4
1880 I=0
1890 I=I+1
1900 :
1910 IF IN$(I)<>"*" THEN 1890
1920 I$=STR$(I)
1930 I$=MID$(I$,2)+LEFT$(LZ$,LZ-LEN(I$)+
1)
1940 IN$(I)=I$+NN$
1950 RN$=STR$(I)
1960 GOSUB 14000:REM  SATZNR.AUFTEILEN
1970 GOSUB 8000:REM  SPEICHERN
1980 IF I>=MX THEN PRINT "□ DATEI VOLL":
GOTO 11500
1990 RETURN
2000 REM   ----------------------------
-
2010 REM  - AUSGABE  1 DATENSATZ
2020 REM  -----------------------------
2030 :
2040 PRINT "□■   ANZEIGE DATENSATZ";RN+
1
2050 PRINT :PRINT :PRINT :PRINT
2060 PRINT " NACHNAME    "NN$
2070 PRINT " VORNAME     "NV$
2080 PRINT " STRASSE     "SR$
2090 PRINT " PLZ         "PL$
2100 PRINT " ORT         "OT$
2110 PRINT " TELEFON     "TE$
2120 PRINT :PRINT :PRINT
2130 RETURN
2140 :
2500 REM ----------------------------
2510 REM    DRUCKEN
2520 REM ----------------------------
2530 :
2540 PRINT "□□ ■  NOCH NICHT DEFINIERT
  ":PRINT
2550 FOR J=1 TO 500:NEXT
2560 RETURN
2570 REM ---------------------------
3000 REM  SUCHEN SEQ/REL
3010 REM ---------------------------
3020 N$=""
3030 PRINT "□":PRINT :PRINT
3040 PRINT "      SUCHEN"
3050 PRINT :PRINT
3060 INPUT " NACHNAME";N$
3070 N=LEN(N$)
3080 S1=1
3090 FOR I=S1 TO MX
3100 :IF IN$(I)="*" THEN 3120
3110 :IF MID$(IN$(I),5,N)=N$ THEN 3190
3120 NEXT I
3130 IF S1>1 THEN PRINT "□■ SUCHE BEENDE
T■":GOTO 3150
3140 PRINT :PRINT N$" NICHT GEFUNDEN"
3150 PRINT :PRINT "DRUECKE TASTE"
3160 GET R$:IF R$="" THEN 3160
3170 RETURN
3180 RN$=LEFT$(IN$(I),4)
3190 GOSUB 14000:REM  SATZNR.AUFTEILEN
3200 GOSUB 9000:REM  LESEN
3210 GOSUB 5000:REM  AUFTEILEN
3220 GOSUB 2000:REM  ANZEIGEN
3230 PRINT :PRINT
3240 W$="                     WAEHLE  "
3250 PRINT "(W)EITERSUCHEN    (Z)URUECK"
3260 PRINT "(A)ENDERN         (L)OESCHEN"
3270 PRINT "(D)RUCKEN                    "
3280 PRINT W$
3290 GET R$:IF R$="" THEN 3290
3300 W$="                     WARTEN  "
3310 PRINT "□ "W$"■"
3320 IF R$="Z" THEN 3170
3330 IF R$="W" THEN S1=I+1:GOTO 3090
3340 IF R$="A" THEN GOSUB 11430:GOTO 324
0
3350 IF R$="L" THEN GOSUB 3670:GOTO 3220
3360 IF R$="D" THEN GOSUB 2500
3370 PRINT "□□□□";:GOTO 3240
3500 REM ----------------------------
3510 REM - LESEN GESAMTE DATEI
3520 REM ----------------------------
3530 RN=0
3540 RN=RN+1
3550 :HB=INT(RN/256)
3560 :LB=RN-HB*256
3570 :GOSUB 9000:REM  LESEN
3580 :IF ER=50 THEN PRINT "□   DATEI END
E ■■■":GOTO 3620
3590 :IF F=2 THEN PRINT "■■ NICHT BELEGT
:  DATENSATZ-NR. ";RN;"■  ":GOTO 3540
3600 :GOSUB 5000:REM  AUFTEILEN
3610 :GOSUB 2000:REM  ANZEIGEN
3620 :PRINT "DRUECKE TASTE"
3630 :GET R$:IF R$="" THEN 3630
3640 :R$=""
3650 IF ER<>50 THEN 3540
3660 RETURN
4000 REM -----------------------------
4010 REM  LOESCHEN  DATENSATZ
4020 REM -----------------------------
4030 :
4040 IN$(I)="*"
4050 RC$="π"
4060 GOSUB 8000:REM  SPEICHERN
4070 NN$=".":NV$=".":OT$=".":TE$=".":PL$
=".":SR$="."
4080 :
4100 RETURN
5000 REM -----------------------------
5010 REM  AUFTEILEN DATENSATZ IN FELDER
5020 REM -----------------------------
5030 :
5050 NN$=MID$(RC$,1,15)
5060 NV$=MID$(RC$,16,15)
5070 SR$=MID$(RC$,31,20)
5080 PL$=MID$(RC$,51,4)
5090 OT$=MID$(RC$,55,15)
5100 TE$=MID$(RC$,70,12)
5110 RETURN
6000 REM -----------------------------
6010 REM - EINGABE NEUE DATEN         -
6020 REM -----------------------------
6030 :
6050 PRINT "■"
6060 PRINT "    EINGABE             "
6070 PRINT :PRINT :PRINT :PRINT
6080 INPUT " NACHNAME ";NN$:NN$=LEFT$(NN
$,15)
6090 INPUT " VORNAME  ";NV$:NV$=LEFT$(NV
$,15)
6100 INPUT " STRASSE  ";SR$:SR$=LEFT$(SR
$,20)
6110 INPUT " PLZ      ";PL$:PL$=LEFT$(PL
$,4)
6120 INPUT " ORT      ";OT$:OT$=LEFT$(OT
$,15)
6130 INPUT " TELEFON  ";TE$:TE$=LEFT$(TE
$,12)
6140 PRINT :PRINT
6150 PRINT "  ADRESSE OK (J/N) ?"
6160 GET R$:IF R$="" THEN 6160
6170 IF R$="N" THEN 6050
6175 IF R$<>"J" THEN 6160
6180 RETURN
6190 :
7000 REM ---------------------------
7010 REM      VERKETTEN DER FELDER  -
7020 REM ---------------------------
7030 :
7040 BL$="                    "
7050 RC$=NN$+LEFT$(BL$,15-LEN(NN$))
7060 RC$=RC$+NV$+LEFT$(BL$,15-LEN(NV$))
7070 RC$=RC$+SR$+LEFT$(BL$,20-LEN(SR$))
7080 RC$=RC$+PL$+LEFT$(BL$,4-LEN(PL$))
7090 RC$=RC$+OT$+LEFT$(BL$,15-LEN(OT$))
7100 RC$=RC$+TE$+LEFT$(BL$,12-LEN(TE$))
7110 RETURN
7120 :
8000 REM  ---------------------------
8010 REM  - SPEICHERN DATEN AUF DISK -
8020 REM  ---------------------------
8030 :
8080 PRINT# 15,"P"+CHR$(2)+CHR$(LB)+CHR$
(HB)+CHR$(1)
8100 PRINT# 1,RC$
8110 FS=1:REM  FLAG FUER SPEICHERN
8170 RETURN
8180 :
9000 REM  ---------------------------
9010 REM  - LESEN DATENSATZ VON DISK -
9020 REM  ---------------------------
9030 F=0
9040 PRINT# 15,"P"+CHR$(2)+CHR$(LB)+CHR$
(HB)+CHR$(1)
9050 INPUT# 15,ER
9060 IF ER=50 THEN 9110
9070 INPUT# 1,RC$
9080 IF RC$<>"π" THEN F=1:GOTO 9110
9090 F=2:REM  FREIER DATENSATZ
9100 :
9110 RETURN
10000 REM  ---------------------------
10010 REM  - DISKETTENFEHLER         -
10020 REM  ---------------------------
10030 PRINT "□"
10040 INPUT# 15,A1,A2$,A3,A4
10050 IF A1=0 THEN 10180
10060 IF A1=62 THEN GOSUB 10200:GOTO 101
80
10070 PRINT
10080 PRINT A1,A2$;A3;A4
10090 PRINT :PRINT
10100 PRINT "        DISKETTENFEHLER"
10110 PRINT :PRINT
10120 PRINT "    BEHEBEN SIE DEN FEHLER
"
10130 PRINT "        UND DRUECKEN SIE"
10140 PRINT
10150 PRINT "             >> F <<"
10160 GET R$:IF R$="" THEN 10160
10170 PRINT "□"
10180 RETURN
10190 :
10200 PRINT "□"
10210 PRINT :PRINT :PRINT :PRINT
10220 PRINT " DIE DATEI       "FR$
10230 PRINT
10240 PRINT " ODER            "FI$
10250 PRINT
10260 PRINT " EXISTIEREN NICHT!"
10270 PRINT :PRINT
10280 PRINT "  L = DATENDISK EINLEGEN"
10290 PRINT
10300 PRINT "  N = DATEI NEU ANLEGEN"
10310 GET R$:IF R$="" THEN 10310
10320 IF R$="L" THEN RETURN
10330 IF R$="N" THEN GOTO 11000
10340 GOTO 10310
11000 REM ---------------------------
11010 REM   - NEUE DATEI ANLEGEN
11020 REM   -------------------------
11030 :
11040 PRINT "□":PRINT
```

tielle Datei

```
11050 IF A1=0 THEN 11070
11060 PRINT " "
11070 PRINT "  ACHTUNG, DIE GESAMTE DISK
ETTE WIRD  "
11080 PRINT "  GELOESCHT ! "
11090 PRINT :PRINT
11100 PRINT " N = NEUE DATEI  X = ENDE"
11110 GET R$:IF R$="" THEN 11110
11120 IF R$="X" THEN  CLOSE 1:GOSUB 1500
0:CLOSE 15:END
11130 IF R$<>"N" THEN 11110
11140 PRINT :PRINT "       BITTE WARTEN"
11150 PRINT# 15,"N:RELATIVE DATEI"
11160 CLR :GOSUB 30000:REM  INIT
11170 CLOSE 1:OPEN 1,8,2,FR$+",L,"+CHR$(
DL)
11180 PRINT "WIEVIELE DATENSAETZE SOLL D
IE DATEI "
11190 PRINT "VERWALTEN? ";
11200 INPUT RN$:RN=ABS(INT(VAL(RN$)))
11210 IF RN<=MX THEN 11180
11220 HB=INT(RN/256)
11230 LB=RN-HB*256
11240 PRINT "BITTE WARTEN"
11250 PRINT# 15,"P"+CHR$(2)+CHR$(LB)+CHR
$(HB)+CHR$(1)
11260 PRINT# 1,CHR$(255)
11270 MX=RN
11280 MX$=STR$(RN)
11290 CLOSE 1
11300 :
11310 PRINT "
         "
11320 FOR I=AM+1 TO MX
11330 :IN$(I)="*"
11340 PRINT "           "MX;I,IN$(I)
11350 NEXT I
11360 FI$="@:"+FR$+"INDEX"
11370 CLOSE 3:OPEN 3,8,3,FI$+",S,W"
11380 IN$(0)=MX$
11390 FOR I=0 TO MX:PRINT# 3,IN$(I):NEXT

11400 CLOSE 3
11420 RUN
11430 :
11500 REM ---------------------------
11510 REM  DATEI ERWEITERN
11520 REM ---------------------------
11530 AM=MX
11540 MX=MX+50
11550 PRINT :PRINT "  ERWEITERN DER DATE
I"
11560 PRINT :PRINT "  BISHERIGE GROESSE=
 "AM
11570 PRINT :PRINT "  NEUE GRØESSE      =
 "MX
11580 RN=MX
11590 GOTO 11220
12000 REM -----------------------------
12010 REM  AENDERN  DATENSATZ
12020 REM -----------------------------
12030 GOSUB 2000:REM  ANZEIGE DATENSATZ
12040 GOSUB 6000:REM  EINGABE NEUE DATEN
12050 LZ$="    ":LZ=4
12060 I$=STR$(I)
12070 I$=MID$(I$,2)+LEFT$(LZ$,LZ-LEN(I$)
+1)
12080 IN$(I)=I$+NN$
12090 GOSUB 7000:REM  VERKETTEN
12100 GOSUB 8000:REM  SPEICHERN
12110 RETURN
12120 :
14000 REM  ---------------------------
14010 REM  - AUFTEILEN DATENSATZNUMMER
14020 REM  ---------------------------
14030 :
14080 RN=ABS(INT(VAL(RN$)))
14100 HB=INT(RN/256)
14110 LB=RN-HB*256
14130 RETURN
14140 RETURN
14150 :
15000 REM ----------------------
15010 REM  SPEICHERN SEQ DATEI  -
15020 REM ----------------------
15030 IF FS<>1 THEN 15120
15040 CLOSE 3:OPEN 3,8,3,"@:"+FI$+",S,W"
15050 GOSUB 10000:REM  FEHLERKANAL
15060 IF A1<>0 THEN 15040
15070 FOR I=0 TO MX
15080 :PRINT# 3,IN$(I)
15090 :PRINT ""I;MX;IN$(I)
15100 NEXT I
15110 CLOSE 3
15120 RETURN
15130 :
15500 REM ----------------------
15510 REM  LESEN     SEQ DATEI  -
15520 REM ----------------------
15530 CLOSE 3:OPEN 3,8,3,"@:"+FI$+",S,R"
15540 GOSUB 10000:REM  FEHLERKANAL
15550 IF A1<>0 THEN 15530
15560 FOR I=1 TO MX
15570 :PRINT# 3,IN$(I)
15580 :PRINT " "I;MX;IN$(I)
15590 NEXT I
15600 CLOSE 3
15610 RETURN
15620 :
30000 REM ---------------------------
30010 REM  INITIALISIERUNG
30020 REM ---------------------------
30030 :
30040 DL=82:REM  DATENSATZLAENGE
30050 RN=1
30060 CLOSE 15:OPEN 15,8,15
30070 BL$="                     "
30080 BL=LEN(BL$)
30090 PRINT "":POKE 53281,0:POKE 53280,
0
30100 FR$="ADR.REL"
30110 FI$=FR$+"INDEX"
30120 DIM IN$(2000)
30130 RETURN
30140 STOP

READY.
```

Listing Index-sequentielle Datei (Schluß)

Liste der verwendeten Variablen

DL	= Datensatzlänge
RN	= Record-Nummer
RN$	= Record-Nummer
BL$	= Leerstellen (Blanks)
BL	= Anzahl Leerstellen
FR$	= Name der relativen Datei
FI$	= Name der Index-Datei
A1 bis A4	= Fehlermeldung der Floppy
IN$(i)	= Inhalt der Index-Datei
IN$(0)	= Größe der Datei
MX/MX$	= Größe der Datei
AM	= Größe der Datei vor Erweiterung
HB/LB	= High-Low-Byte
ER	= Fehlermeldung der Floppy bezogen auf relative Datei
RC$	= ein gesamter Datensatz
NN$	= Nachname
NV$	= Vorname
SR$	= Straße
OT$	= Ort
PL$	= Postleitzahl
TE$	= Telefon
ST	= Statusvariable (prüft auf Dateiende bei der Sequent. Datei)
Sl	= stellt fest, ob die gesamte Datei durchsucht wurde
F	= stellt fest, ob ein Datensatz belegt ist
FS	= wenn 1, dann wird vor Beenden des Programms die sequentielle Datei erneut abgespeichert.

Zeile 9070 INPUT#1,RC$ durch folgende Zeilen zu ersetzen:
9068 RC$=" "
9070 GET#1,W$
9072 RC$=RC$+W$
9074 IF W$=CHR$(255) THEN 9090
9076 IF W$=CHR$(13) THEN 9070

Den GET#-Befehl kann man natürlich auch dann einsetzen, wenn die Satzlänge kleiner als 88 Zeichen ist. Dann werden die Zeichen jedoch um einiges langsamer eingelesen als mit dem INPUT#-Befehl. Aber es ist mit dem GET#-Befehl möglich, bis zu 255 Zeichen in eine Variable einzulesen.

Ein Problem ganz anderer Art kann auftauchen, wenn Sie eine Floppy besitzen, die vor Dezember 1983 gekauft beziehungsweise hergestellt wurde und gleichzeitig mit einem VC 1526 drucken. Es kann möglich sein, daß bei Benutzung einer relativen Datei Fehler auftauchen, sobald dieser Drucker eingeschaltet ist. Probieren Sie das vorher aus! Abhilfe schafft meines Wissens nur ein neues DOS für die alte VC 1541.

Anregungen für Programmierer

Dieses Programm ist — bis auf die Druckerroutine, die sich jeder für seinen eigenen Drucker selber schreiben sollte — komplett und funktionsfähig. Sicherlich lassen sich noch eine ganze Reihe von Schönheitsreparaturen machen. Auch eine Erweiterung des Programms ist durchaus sinnvoll. Zum Beispiel könnte man ein Unterprogramm schreiben, das, ähnlich einem professionellen Dateiprogramm, es ermöglicht, sich die Datenfelder selbst zu definieren. Auch die Suche nach mehr als einem Feld ist durchaus sinnvoll. Wenn erwünscht, kann auch eine Sortierroutine eingefügt werden. Der modulare Aufbau erlaubt dies ohne Schwierigkeiten. So kann sich jeder, der etwas Programmiererfahrung hat, eine Dateiverwaltung aufbauen, die sogar professionellen Programmen etwas voraus hat: Sie ist auf die persönlichen Bedürfnisse abgestimmt und läßt sich auch jederzeit ändern. Wer das Programm kompiliert, kann auch bei sehr großen Dateien noch mit guten Suchgeschwindigkeiten rechnen. Dieses Beispielprogramm enthält alle Voraussetzungen für eigene Erweiterungen. Ich selbst habe obengenannte Funktionen für meine eigene Dateiverwaltung bereits realisiert. (gk)

Hard and Soft: Eine Kleine

Von der Qualität und leichten Bedienbarkeit der Programme hängt die Qualität der Keyboardarrangements immer mehr mit ab. Nichts-desto-trotz sollte man immer daran denken, daß ein schlechter Song auch mit der ausgefeiltesten Software nicht besser wird. Hier ein kurzer Überblick über Midi-Software und -Interfaces für den Commodore 64.

Steinberg Research: 16-Spur-Midi-Recorder, Interface und Drum to Midi Converter

Vom Keyboarder für Keyboarder entwickelt wurde die Steinberg-Midi-Software: Einer der beiden Entwickler ist selbst Keyboarder in der Gruppe um die Rock-Lady Inga Rumpf. Dieselbe Firma verteibt auch ein Mini-Midi-Interface mit einem Midi-Input und zwei Outputs. Das Interface selbst besteht aus einer Platine mit festgelöteten Buchsen. Die Platine wird direkt in den User-Port des C 64 gesteckt. Leider hat man, wahrscheinlich aus Kostengründen, auf ein Gehäuse verzichtet. Hier empfiehlt es sich, auf jeden Fall selbst Hand anzulegen. Preis zirka 120 Mark.

Die Software kann man als 16 Spur-Multitrack-Recorder bezeichnen. 16 Sequenzen verschiedener Länge haben im Arbeitsspeicher Platz. Die einzelnen Sequenzen spielt man Spur für Spur ein. Jede faßt bis zu 16 polyphone Spuren und unterschiedliche Parameter. Pitchbending und Modulation, Dynamik und After Touch sowie Sound-Änderungen werden mit aufgezeichnet. Natürlich nur, wenn das Instrument dazu in der Lage ist. Jede Spur kann dabei so viele Stimmen aufnehmen, wie das Einspielkeyboard zur Verfügung stellt. Längere Kompositionen bildet man durch Verknüpfen der 16 Sequenzen, wobei die Reihenfolge frei wählbar ist. Die 16 Sequenzen und 16 Spuren erscheinen recht musikerfreundlich in einer Art Songtable am Bildschirm. Man arbeitet ausschließlich mit diesem Bild (Bild 1). Bereits während der Aufnahme werden etwaige Timingfehler korrigiert, wobei die Korrektur für jede Spur individuell anwählbar ist. (Korrektur auf $\frac{1}{4}$-, $\frac{1}{8}$-, $\frac{1}{16}$-, $\frac{1}{32}$- und $\frac{1}{64}$-Werte möglich). Ein Metronom hilft während des Einspielens, das richtige Tempo zu halten.

Bis zu 16 Midi-Instrumente spricht das Interface im Playmode an. Die 16 Recorder-Spuren lassen sich natürlich beliebig auf die 16 Channels und somit verschiedenen Instrumente verteilen. Die Software ist für OMNI-, POLY- und MONOmode ausgelegt.

Ein Farbbildschirm ist unbedingt nötig. Die einzelnen Betriebsmodes, wie Aufnahme, Play und so weiter, erkennt man durch verschiedene Hintergrundfarben. Bespielte Spuren lassen sich in jede beliebige Sequenz und dort an jeden Platz kopieren, sowie in einem Bereich von +32 bis +32 Halbtönen transportieren. Preis 290 Mark.

Für schwierige Synchronisations-Aufgaben in größeren Midi-Systemen, stellt Steinberg eine auf die Midi-Multitrack-Recorder-Software und den C 64 abgestimmte Synchronisier-Platine her. Mit dieser läßt sich dann eine Band-Maschine synchronisieren (Tape Sync) oder der Midi-Recorder extern triggern. Umgekehrt kann man ihm diverse Clock-Signale und einen Start-Impuls zur Steuerung externer, noch nicht Midi-kompatibler Elektronik-Drums entnehmen. Preis zirka 98 Mark.

Demnächst erscheint im Programm von Steinberg ein Drum-to-Midi-Converter. Dies wäre das erste Gerät dieser Art. Mit diesem Gerät kann man dann endlich Percussion-Impulse direkt in die Midi-Software einspielen. Hierzu ist zusätzlich Hardware nötig. Die Impulse können entweder von einem Pad-Set (Simmons oder ähnliches) oder über Mikrofon von einem »echten« Schlagzeugset abgenommen werden.

Jellinghaus Music Systems: Midi-Interfaces und Software

Jellinghaus, einer der deutschen Pioniere auf dem Gebiet der Midi-Technik, bietet zwei verschiedene Interface-Versionen an. Ein sogenanntes Mini-Interface, zum Preis von 115 Mark, daß sich ausschließlich an den Commodore 64 anschließen läßt, sowie eines mit mehr Features, das sich sowohl mit 6502 als auch Z80-Prozessoren ansprechen läßt, zum Preis von 330 Mark. Das Mini-Interface verfügt lediglich

MARKTÜBERSICHT

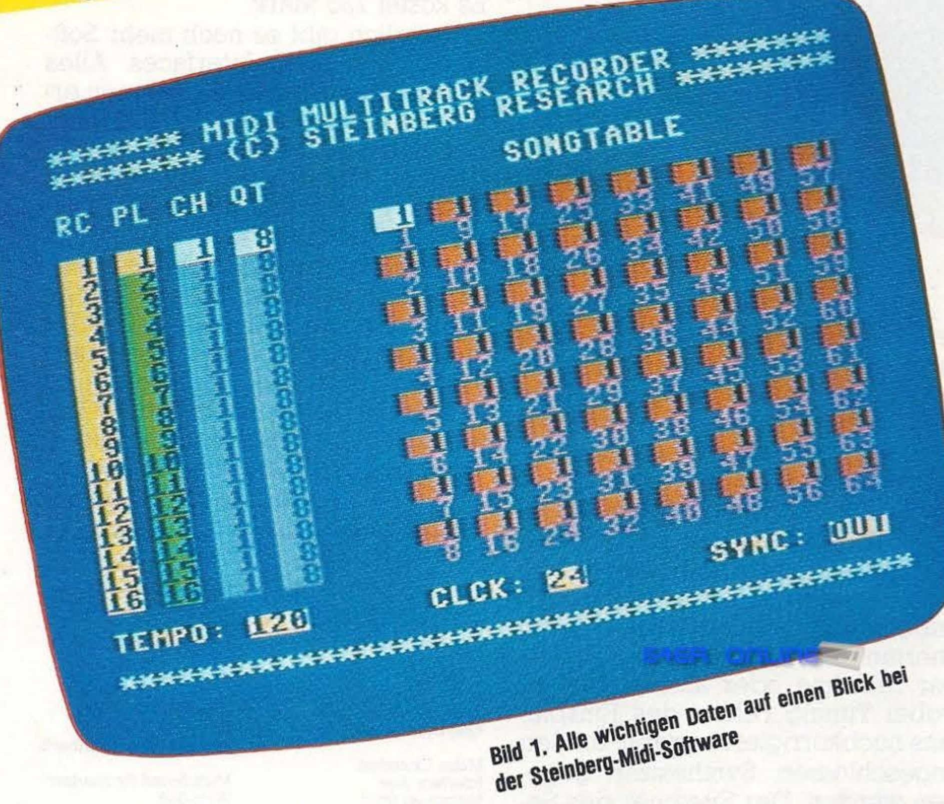


Bild 1. Alle wichtigen Daten auf einen Blick bei der Steinberg-Midi-Software

über einen In- und zwei Outputs. Die größere Version bietet zusätzlich eine Midi-Thru-Buchse sowie Drum Sync-Möglichkeit.

Jellinghaus bietet diverse Software für den Commodore 64 an. Vor allem Yamaha-DX-7-Besitzer kommen hier auf ihre Kosten. Der Sound-Editor DX-7/DX-9 zeigt alle Soundparameter dieser Keyboards übersichtlich auf dem Bildschirm an. Dies weiß jeder zu schätzen, der sich schon an der Programmierung der beiden Keyboards versucht hat. Die einzelnen Parameter lassen sich nun bequem über die C 64-Tastatur editieren und anschließend ausdrucken.

Überdies entkommt man auf diese Weise auch den teuren RAM-Cartridges, denn mit dieser Software lassen sich sämtliche Sounddaten auch auf die Commodore-Diskette speichern. Preis zirka 185 Mark.

Auch eine Multitracker-Software gibt es hier, den sogenannten Multitrack Live-Sequenzer für den Commodore 64 (Bild 2).

Er stellt 12 Spuren zur Verfügung, natürlich wieder voll polyphon. 10000 Events (note on/note off) haben insgesamt im Speicher Platz. Ein Metronom sorgt für den richtigen Takt, die Aufnahme startet mit einem wählbaren Ereignis, zum Beispiel der ersten gespielten Note, einem Druck auf die Programmwechseltaste oder durch Drehen am Pitch-Bender. Für jede Aufnahme-Spur läßt sich getrennt festlegen, welche Parameteränderungen gespeichert werden sollen, zum Beispiel Keyboarddaten, Anschlagsdynamik, Programmwechsel, Pitch Bender und andere. Die Auswahl erfolgt in einem Filter-Menü. Diese Bezeichnung erscheint mir hier allerdings etwas fehl am Platze. Beim Arbeiten mit einer Drum-Box kann entweder diese den Recorder, oder der Recorder diese synchronisieren. Das Tempo läßt sich im Bereich von 40 bis 200 regeln, die Taktart kann von $\frac{2}{2}$ bis $\frac{11}{2}$, $\frac{2}{4}$ bis $\frac{11}{4}$, $\frac{2}{8}$ bis $\frac{11}{8}$ gewählt werden. Natürlich auch hier alle drei Midimodes und wählbare Zuordnung von Spuren auf Channels. Einzigartig bisher: Die gespeicherten Songs lassen sich listen und editieren. Auf dem Bildschirm erscheint hierbei ein korrektes Zeitprotokoll der Reihenfolge, in der bestimmte Tasten gedruckt und wieder losgelassen wurden, mit Angabe der zusätzlich aufgenommenen Parameter. Außerdem lassen sich alle Spuren nachträglich im Timing korrigieren, in $\frac{1}{4}$ bis $\frac{1}{32}$-Werten, sowie $\frac{1}{4}$- bis $\frac{1}{32}$-Triolen. Weitere Features: Endlos-Wiedergabe (loops), Fuß-Schalter-Anschluß, Transponierung und Loudness-Skalierung jedes Tracks und die Möglichkeit, mehrere Tracks auf einen abzumischen (Mix-Down). Das Jellinghaus Midi-Recording-Studio kostet 250 Mark.

Eine weitere interessante Midi-Software: das Master-Keyboard (Bild 3). Dieses Programm ist interessant, wenn man viele Instrumente an seinem Midi-System angeschlossen hat und live darauf spielen will. Die Einspielklaviatur läßt sich dann in verschiedenen Weisen zur Steuerung der anderen Synthis einsetzen. So lassen sich zum Beispiel auf dem Einspiel-Keyboard (Master Keyboard) sechs Splitpunkte bestimmen. Mit den so entstandenen Klaviaturabschnitten kann man dann die restlichen Synthis gezielt vom Master-Keyboard aus live spielen. Außerdem können für die angeschlossenen Keyboards oder Effektgeräte 80 Presets programmiert werden, so daß sie bei Anwahl eines dieser Presets durch einen Tastendruck auf die bestimmten Klangbeziehungsweise Effektprogramme geschaltet werden. Ein drittes Feature ermöglicht zu jedem gespielten Ton andere hinzumischen. Diese Software kostet 200 Mark.

Passport Design: Midi-Interface und Software

Passport Design ist in Computermusik-Kreisen durch ihr System für den Apple II, das Mountain Board Music System, wohlbekannt. Mittlerweile wurde auch Midi-Software und ein Interface für den C 64 von dieser Firma entwickelt. Auf der Midi-Interface-Karte sind drei 5-Pol-DIN-Buchsen vorhanden. Einmal Midi-In, einmal Midi-Out und eine dritte Buchse, für die Synchronisa-

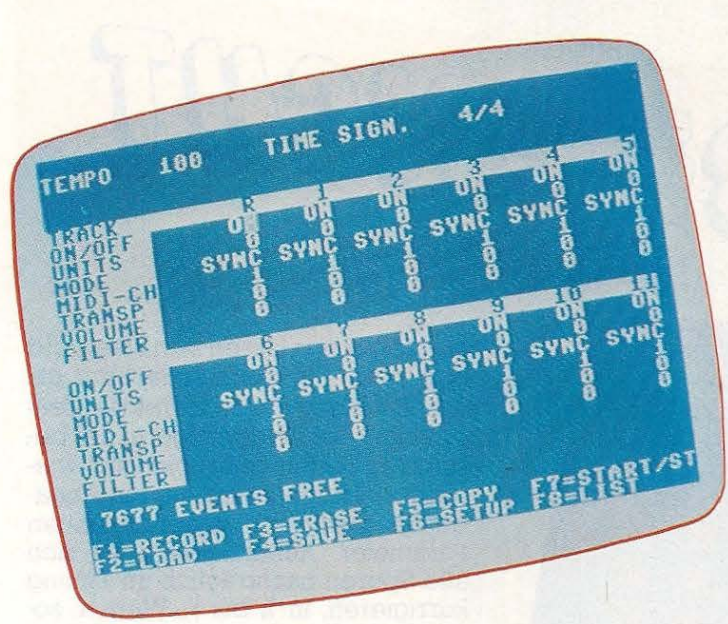


Bild 2. Das Hauptmenü beim Multitrack Live-Sequenzer

tion einer Drum-Maschine (Drum Sync). Die Midi-Interfacecard überträgt und empfängt sämtliche Standard-Midi-Daten. Sie kostet in Deutschland 590 Mark.

Mit dem Midi-Recorder Midi/4 kann man bis zu 16 Stimmen Real-Time einspielen, beliebig über vier Aufnahmespuren verteilt. Hierbei speichert die Software alle für die Komposition wichtigen Informationen, also Tonhöhe, Dauer, Anschlagsdynamik, Pitch-Bend, Presetänderungen und After Touch. Sollte man sich einmal verspielt haben, können einzelne Stellen mit der »Punch-In«-Funktion während des Abspielens korrigiert werden, — so, als hätte man eine der guten alten Vier-Spur-Bandmaschinen vor sich. Natürlich lassen sich alle Midi-kompatiblen Rhythmusmaschinen synchronisieren. Auch Geräte ohne Midi-Bus, wie zum Beispiel ältere Electronic-Drums der Firmen Roland und Korg, kann man anschließen, sofern sie einen 5-Pol-DIN-Stecker zur Synchronisation besitzen. Das Schlagzeug wird durch die Software gestoppt und gestartet. Weitere Features von Midi/4 sind eine »Loop«-Funktion, »Clicktrack on/off«, die die Synchronisation des Midi-Sequenzers mit einer Bandmaschine erlaubt und »Transposition«. Der Preis beträgt in Deutschland zirka 295 Mark.

Sequential Circuits: Model 64-Sequenzer für den Commodore 64

Der Model 64 Midi-Sequenzer Sequential Circuits ist als Cartridge entwickelt, die in den Memory-Expansion-Port des C 64 gesteckt wird. Um ihn voll ausnutzen zu können, benötigt man ein sechsstimmig polyphones, Midi-fähiges Keyboard. Der Sequenzer zeichnet dann exakt das auf, was von der Tastatur her eingespielt wird. Insgesamt können bis zu 4000 Noten gespeichert werden. Verfügt das benutzte Keyboard über Anschlagsdynamik, so wird auch diese mit aufgezeichnet.

Der Sequenzer merkt sich auch alle Pitch-bend- beziehungsweise Modulationsinformationen. Im Wiedergabemodus können alle gespeicherten Informationen dann entweder real-time oder auto-corrected, wobei Timing Fehler des Einspielens nachkorrigiert werden, an den angeschlossen Synthesizer gegeben werden. Der Speicher des Sequenzers läßt sich in acht Blocks unterteilen, jeder dieser Blocks enthält dann eine sechsstimmig polyphone Sequenz, die alle unterschiedliche Längen haben können. Die Sequenzen kann man nachträglich per Software ganz, oder in Teilbereichen ändern, transportieren und auf Diskette beziehungsweise Kassette abspeichern. Der Sequenzer ist so konstruiert, daß er auch ohne Monitor betrieben werden kann. LEDs auf der Frontplatte signalisieren den jeweiligen Betriebszustand, was natürlich vor allem für Livemusiker auf der Bühne praktisch ist. An den Sequenzer kann man einen Fußschalter anschließen, zum Starten und Stoppen, wenn keine Hand frei ist; außerdem läßt er sich mit externen Rhythmusgeräten synchronisieren. Es kostet 725 Mark.

Natürlich gibt es noch mehr Software, noch mehr Interfaces. Alles Aufzuzählen würde den Rahmen erheblich sprengen. Für den Keyboarder zumindest, kann ein gut durchdachtes Midi-System mit entsprechender Software ein herkömmliches Recordingsystem mit Mehrspurmaschine und Mischpult in vielen Fällen ersetzen. Billiger kommt man jedoch auch nicht weg. Die Anschaffungskosten eines Computersystems und der Midi-Soft- und Hardware dürften sich in der Größenordnung eines Acht-Spur-Recorders der Low-Cost-Klasse bewegen.

(Richard Aicher/aa)

Synthesizer 64
Werner Kracht
Efteloh-Weg 38
2000 Hamburg 51

Musicalc
Lucius Computer-Programme
Theodor-Körner-Str. 5
4220 Duislaken 1

Music Extended
Interface Age
Vohburger Str. 1
8000 München 21

Synthimat, Synthy 64
Data Becker
Merowinger Str. 30
4000 Düsseldorf 1

Musik Composer
Commodore

Colortone Keyboard
Waveform Co.
1912 Bonitaway
Berkley CA 94704
USA

Microsound Keyboard
Autographics Limited
3a Reading Road
Henley on Thames
Oxon
RG9 1AB

Ultisynth
Micro Händler
4050 Mönchen Gladbach

Multi Sound Synthesizer
Wico-Soft
Nordstr. 22
3443 Herleshausen

Jellinghouse
Martineer Hellweg 40
4600 Dortmund 70

Sequential Circuits
PO 16
3640 AA Mijdrecht
Holland

Passport Design
Roland Späth
Colmarer Str. 16
7000 Stuttgart 40

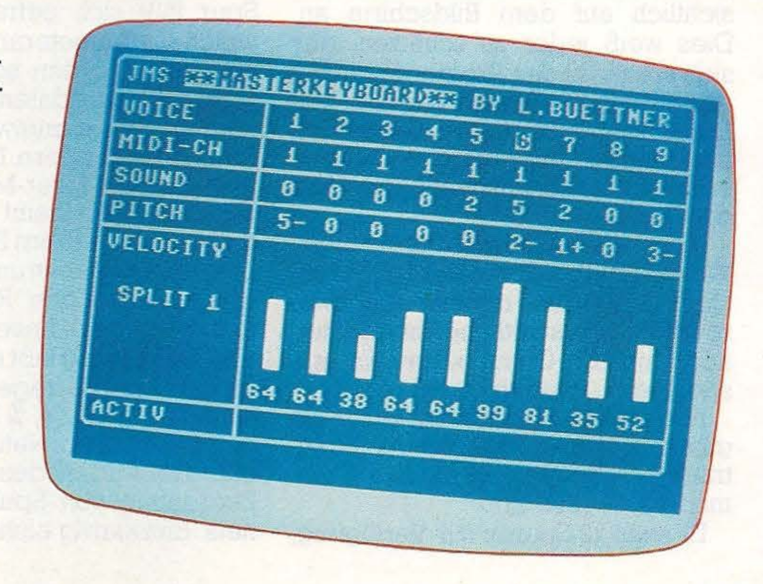


Bild 3. Bildschirmdarstellung beim Master-Keyboard


64ER ONLINE

KLANGSYNTHESE UND

Unser Commodore 64 ist ein hervorragender Musikus. Manches Musikpro Will man ihm selbst gezielt Töne entlocken, muß man über einige grundle

Drei verschiedene Kurvenformen und sogenanntes »weißes Rauschen« stellt der Sound-Chip des Commodore 64 für unsere Experimente zur Verfügung. Die drei Kurvenformen nennt man Dreieck, Rechteck und Sägezahn (siehe Bild 1).

Jede der drei Kurven klingt ganz charakteristisch und zwar unabhängig von der Tonhöhe. Keine der drei klingt identisch mit irgendeinem natürlichen Instrument. Leider, sonst wäre nämlich alles viel einfacher. Bei den genannten Schwingungsformen handelt es sich um sehr einfache Basiskurven. Betrachtet man Kurvenformen natürlicher Klangerzeuger, also etwa eines Musikinstruments oder irgend eines anderen Geräusches am Oszillograph, stellt man sehr viel kompliziertere Kurvenverläufe fest (siehe Bild 2).

Außerdem verändern sich die Kurven kontinuierlich während des gesamten Klangablaufes. Wie lassen sich so komplexe Abläufe elektronisch realisieren?

Zu Beginn des 19. Jahrhunderts lebte der französische Mathematiker Jean Baptiste Joseph Fourier. Er entwickelte die mathematisch-physikalischen Grundlagen, die uns auch heute noch Klänge verstehen und elektronisch nachahmen lassen. Es handelt sich um das Verfahren der sogenannten harmonischen Analyse von Klängen. Heute nennen wir das zugrundeliegende mathematische Prinzip ihm zu Ehren Fourier-Synthese. Fourier stellte fest, daß sämtliche periodischen Schwingungen in eine sinusförmige Grundschwingung und ebenfalls sinusförmige Oberschwingungen zerlegt werden können. Man nennt diese auch harmonische Oberschwingungen oder einfach Harmonische. Diese unterscheiden sich nur in der Frequenz und der jeweiligen Amplitude. Er bewies auch, daß die Umkehrung dieses Satzes ebenso gilt.

Jede periodische Schwingung läßt sich durch Addition ihrer Grundkomponenten, also Sinusschwingungen mit bestimmter Frequenz und Amplitude, aufbauen. Sein Beweis war natürlich rein mathematischer Natur. Prinzipiell handelt es sich hier um Mathematik in Reinkultur. Natürlich bestehen nicht alle Klänge aus periodischen Schwingungsverläufen. Deshalb kann man mit diesem Verfahren auch nicht alle Klänge analysieren beziehungsweise synthetisieren. Rauschen ist zum Beispiel eine absolut nicht periodische Schwingungsform und kann nicht nach diesem Verfahren behandelt werden. Für die Frequenzen der Harmonischen

Unterschiedliche Obertongemische produzieren verschiedene Klänge

gilt die Bedingung, daß sie in einem geradzahligen Verhältnis zur Frequenz der Grundschwingung stehen. Die Reihen der möglichen Sinuskurven haben also alle denselben Frequenzabstand, nämlich die Frequenz der Grundschwingung. Die Grundschwingung ist die Sinuskurve mit der tiefsten Frequenz. Sie bestimmt meist gleichzeitig die Tonhöhe des Klangs. Die Harmonischen bestimmen den Charakter des Klanges.

Wie ein Ton nun letztlich klingt, bestimmt die Zusammensetzung seines Obertongemisches. Entscheidend ist, welche Harmonischen beteiligt sind und mit welcher Amplitude. Betrachten wir wieder unsere Grundschwingungsformen. Hier handelt es sich um reine periodische Grundschwingungen. Sie lassen sich auch sehr leicht mit Mitteln der Elektronik erzeugen. Deshalb sind sie für uns so wichtig.

Der Sinuston besitzt keine Oberschwingungen, er ist rein. Sein Klang hört sich unnatürlich weich an. Reine Sinustöne existieren in der Natur nicht. Ganz anders der Sägezahn. In einer Sägezahnschwingung sind alle Obertöne vertreten. Der n-te Oberton ist hierbei 1/n mal so laut wie der Grundton. Die Amplitude der Obertöne nimmt also hier mit zunehmender Ordnungszahl ab. Der Sägezahn klingt hell, trompetenhaft. Die Dreieckschwingung weist sehr viel weniger Obertöne auf. Sie klingt dumpf. Man kann den Klang auch als flötenähnlich bezeichnen. Die Rechteckschwingung besteht aus ungeradzahligen Obertönen. Sie klingt hohl.

Bob Moog konstruierte den ersten Synthesizer

Bob A. Moog hatte als erster die Idee, diese Grundschwingungsformen zur Klangsynthese einzusetzen. Inspiriert von den Klangwünschen seines Freundes Herb Deutsch, ei-

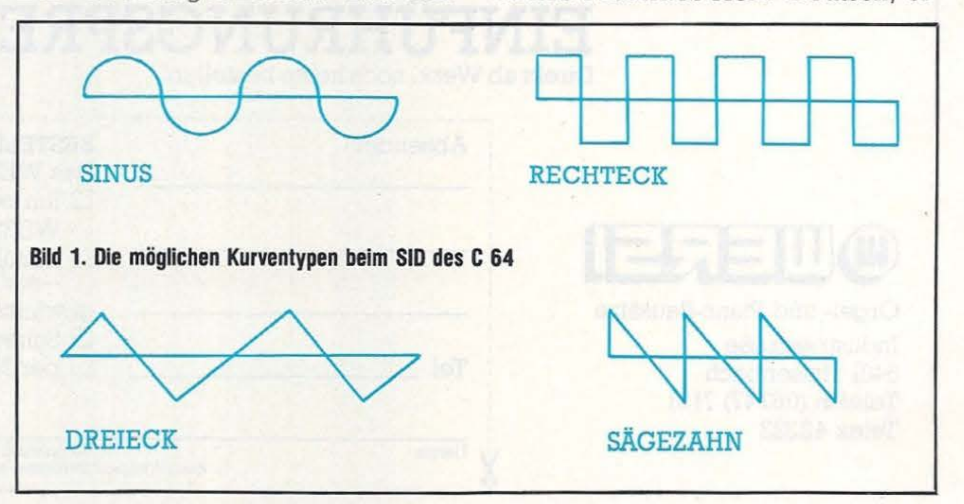


Bild 1. Die möglichen Kurventypen beim SID des C 64

SYNTHESIZERTECHNIK

gramm läßt ihn sogar teurere, professionelle Musiksynthesizer übertreffen. gende Dinge der Klangsynthese und Synthesizertechnik Bescheid wissen.

nes Experimental-Musikers, hatte er den Sommer 1964 damit verbracht, Elektronik-Module zu konstruieren, die die absonderlichsten Klänge hervorbrachten. Mit diesen Modulen war es erstmals möglich, die drei Größen, mit denen sich jedes Klangereignis beschreiben läßt, nämlich Tonhöhe (Frequenz), Klangfarbe (Kurvenform-Oberwellengehalt) und Lautstärke (Amplitude) relativ genau zu kontrollieren. Die entscheidende Idee war die Einführung der Spannungssteuerung. Er entwickelte Module, die direkt diesen drei Größen zuzuordnen und mittels Spannungssteuerung elektronisch kontrollierbar waren.

So entstand der Moog-Synthesizer, ein Klangmonster von 1 x 2 m Größe. Dieses Gerät revolutionierte die gesamte elektronische Tonerzeugung grundlegend. Auch unser kleiner SID-Chip ist prinzipiell nichts anderes als ein Nachfolger des Riesenkastens von einst. Nur mit dem Unterschied, daß er aus wenigen Kubikzentimetern Raum Leistung zaubert, für die ehedem der Moog-Synthesizer Kubikmeter benötigte und statt analoger Technik und Spannungssteuerung Digitaltechnik und Digitalkontrolle einsetzt.

Eines hat jedoch der Moog von einst dem SID-Chip von heute immer noch voraus. Er klingt wesentlich besser. SID-Klänge sind nicht sehr bombastisch. Es fehlen volle Bässe und höchste Höhen. Deshalb konnte sich auch bisher der Commodore 64 mit dem SID bei Profimusikern als Instrument nicht etablieren.

Im folgenden möchte ich kurz die einzelnen Funktionsgruppen eines Synthesizers, wie es auch unser SID ist, beschreiben.

Töne und Melodien entstehen in Tongeneratoren. Unser SID stellt uns drei unabhängig voneinander funktionierende Tongeneratoren zur Verfügung. Jeder produziert drei verschiedene Kurvenformen, nämlich Dreieck, Sägezahn und Rechteck. Überdies erzeugt jeder noch weißes Rauschen, ein Gemisch sämtlicher Frequenzen, die von einem Zufallsgenerator erzeugt werden.

Töne entstehen in Oszillatoren

Wir haben gehört, daß sich unsere Basis-Kurvenformen durch ihren unterschiedlichen Gehalt an Obertönen und somit durch ihren Klang prinzipiell unterscheiden. Nach einigem Üben mit dem SID wissen wir bald, welche Kurvenform sich für einen bestimmten Klang anbietet.

Verschiedene Kurvenformen, schön und gut, aber wozu drei Tongeneratoren? Ganz einfach. Musik besteht nicht nur aus Solo-Melodien. Mit einem Tongenerator alleine könnten wir kaum Musik produzieren. Drei Tongeneratoren lassen uns jedoch schon ein ganzes Kammermusik-Trio realisieren. Das heißt, wollen wir drei verschiedene Töne gleichzeitig erzeugen, also dreistimmig polyphon spielen, muß für jeden Ton ein eigener Tongenerator vorhanden sein.

Der Rauschgenerator — kein Sturm ist ihm heilig

Mit dem Rausch-Generator (Noise-Generator) ist es möglich, Effekte wie Wind und Brandung zu erzeugen, den Klang einer Pauke oder eines Beckens, den Anblaswind von Orgelpfeifen, Holzblasinstrumenten und schließlich Gewehrschüsse in allen Variationen nachzuahmen.

Ein Ton macht nun aber noch lange keine Musik. Viele Töne unterschiedlicher Tonhöhe sind hierzu nötig. Deshalb muß die Tonhöhe der Oszillatoren steuerbar sein. Wie wir schon wissen, erfand Moog das Prinzip der Spannungssteuerung. Sämtliche Module seines Synthesizers konnten von analogen Steuerspannungen geregelt werden. Man nannte deshalb die Module auch »voltage controlled modules« (spannungsgesteuerte Module).

Computer hantieren nur ungern mit analogen Spannungen. Auch die Tongeneratoren unseres SIDs werden deshalb nicht von Analogspannungen, sondern mittels digitalen Informationen gesteuert. Wir nennen diese Oszillatoren deshalb auch nicht VCOs (voltage controlled oscillators), wie dies Moog machte, sondern besser DCOs (digital controlled oscillators).

Musiker wollen mit ihrem Instrument spielen. Es soll Töne produzieren. Wie teilen wir unseren Tongeneratoren mit, welche Töne sie spielen sollen? Routinierte Klavierspieler möchten ihre Ideen am liebsten über eine Klaviatur einspielen. Leider verfügt der Commodore 64 über keine richtige. So bleibt nichts

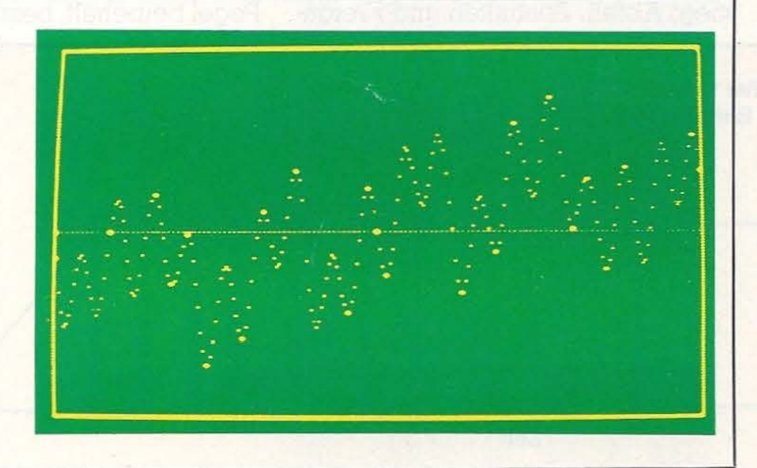


Bild 2. Kurvenform eines natürlichen Klangerzeugers

anderes, als die Alpha-Tastatur in ein Keyboard zu verwandeln. Hierzu ordnet man jeder QWERTY-Taste eine Tonfrequenz zu. Drückt man die Taste, erklingt der Ton. Das Spielen auf einer Schreibmaschinentastatur bereitet jedoch wenig Freude, zumindest muß man sich erst daran gewöhnen.

Kürzlich erschienen jedoch zwei Klaviaturen auf dem Markt, die mit richtigen Klaviertasten ausgerüstet, an den Commodore 64 angeschlossen werden können. Ihre Anschaffung lohnt sich sicher für den, der mit dem Commodore ausschließlich Musik machen will und eine Klaviertastatur gewöhnt ist.

Viele beherrschen das Klavierspiel jedoch nicht. Sie können trotzdem Musik mit dem Commodore 64 komponieren. Die Kompositionen werden nun nicht »Live« eingespielt, sondern Ton für Ton über die Alpha-Tastatur eingetippt. Es spielt dabei keine Rolle, in welcher Geschwindigkeit die Eingabe erfolgt. Man kann sich also viel Zeit lassen und Eingabefehler nachträglich ausbessern. Die einzelnen Töne werden hierbei mittels mehreren Eingaben für Tonhöhe, Tondauer und Lautstärke bestimmt. Das Verfahren ist relativ langwierig. Hat man alles eingegeben, spielt der Commodore den Song in der gewünschten Geschwindigkeit ab.

Bei den meisten ordentlichen Musikprogrammen kann der Computer die eingespielten oder eingetippten Songs speichern und auf Befehl wieder ausgeben. Die Länge der Kompositionen kann hierbei meist einige tausend Töne betragen. Ein Moog-Synthesizer von einst konnte sich nur 16 Töne merken und diese zyklisch immer wieder abspielen. Man nannte Geräte, die dies ermöglichten, Sequenzer. Heute benutzt man diesen Begriff meist allgemein für Tonspeichersysteme.

Wir sagten schon, daß sich jeder Klangeindruck durch die drei Größen Tonhöhe, Klangfarbe und Lautstärke beschreiben läßt. Wir wissen nun, daß die Töne in den Oszillatoren entstehen und die Tonhöhe durch digitale Steuerung dieser Oszillatoren entsteht. Außerdem können wir durch Auswahl der Kurvenformen bereits unterschiedliche Klangfarben erhalten. Wie können wir nun die Lautstärke bestimmen?

Der ADSR steuert den Lautstärkeverlauf der Klänge

Der Klang eines Klaviers weist ein ähnliches Frequenzspektrum wie der einer Geige auf. Trotzdem unterscheiden sich die beiden Klänge voneinander gravierend. Eine wesentliche Ursache hierfür ist der unterschiedliche zeitliche Verlauf der Lautstärken. Schlägt man eine Klaviersaite an, setzt der Ton schlagartig ein und schwillt dann relativ rasch wieder ab. Im Gegensatz dazu schwillt der Geigenklang langsam an und behält, solange der Bogen gestrichen wird, eine konstante Lautstärke bei. Erst wenn der Bogen nicht mehr über die Saite streicht, klingt der Ton langsam ab. Klavier und Geige unterscheiden sich also im zeitlichen Verlauf der Lautstärke in der sogenannten »Hüllkurve«.

Jedes Instrument, jedes Geräusch zeichnet sich vor allem durch einen ganz spezifischen Lautstärkenverlauf aus. Hier spielen sich sehr komplexe Vorgänge ab, und nicht zuletzt bestimmt die Möglichkeit, solche komplexe Hüllkurven nachzubilden, die Qualität eines Synthesizers. Moog reduzierte bei der Entwicklung seines Synthesizers den Verlauf der Hüllkurve auf die Nachbildung der für das menschliche Klangempfinden wichtigsten vier Phasen: Attack, Decay, Sustain und Release (ADSR). Zu Deutsch: Anstieg, Abfall, Aushalten und Freigabe des Lautstärkeverlaufes. Der sogenannte ADSR- oder Hüllkurvengenerator liefert nun eine dem gewünschten Hüllkurvenverlauf entsprechende Steuerinformation. In unserem SID handelt es sich hierbei wieder um Digitalinformationen, die direkt die Lautstärke der drei Tongeneratoren beeinflussen. In Analogsynthesizern benutzt man auch in diesem Fall Steuerspannungen. Unser SID stellt drei ADSRs zur Verfügung. Jede Stimme kann folglich einen eigenen Hüllkurvenverlauf erhalten. In Bild 3 sehen wir links eine typische ADSR-Kurve. Auf der x-Achse ist die Zeit, auf der y-Achse die Amplitude (Lautstärke) des Klanges abgetragen. Sofort nach dem Anschlag einer Taste würde die Lautstärke des Tons zunächst langsam bis zum Maximum ansteigen (Attack) und dann auf einem bestimmten Sustainpegel einige Zeit verweilen (Sustain). Läßt man die Taste los, verklingt er in der Abklingphase (Release), bis man nichts mehr hört. Natürliche Attack-Werte liegen, je nach Klang, im Bereich von einigen Millisekunden bis zu einigen Sekunden. Kurze Attack-Werte besitzen alle percussiven Instrumente wie Pauke, Schuß, auch Saiten-Instrumente wie Gitarre und Klavier gehören hierzu. Längere Attack-Zeiten finden wir zum Beispiel bei Blasinstrumenten. Es dauert, bis die Atemluft des Trompeters das Instrument zum Klingen bringt. Streichinstrumente weisen noch längere Attack-Werte auf. Sehr lange Attack-Zeiten, im Bereich vieler Sekunden, benötigt man zum Beispiel, um Wind oder Brandungsgeräusche nachzuahmen. Hat der Ton dann den maximalen Attack-Pegel mehr oder weniger schnell erreicht, können wir ihn in der ebenfalls regelbaren Decay-Phase bis auf einen gewissen Pegel abfallen lassen, den Sustain-Pegel.

Wie lange der Ton den Sustain-Pegel beibehält, bestimmt das Gate-

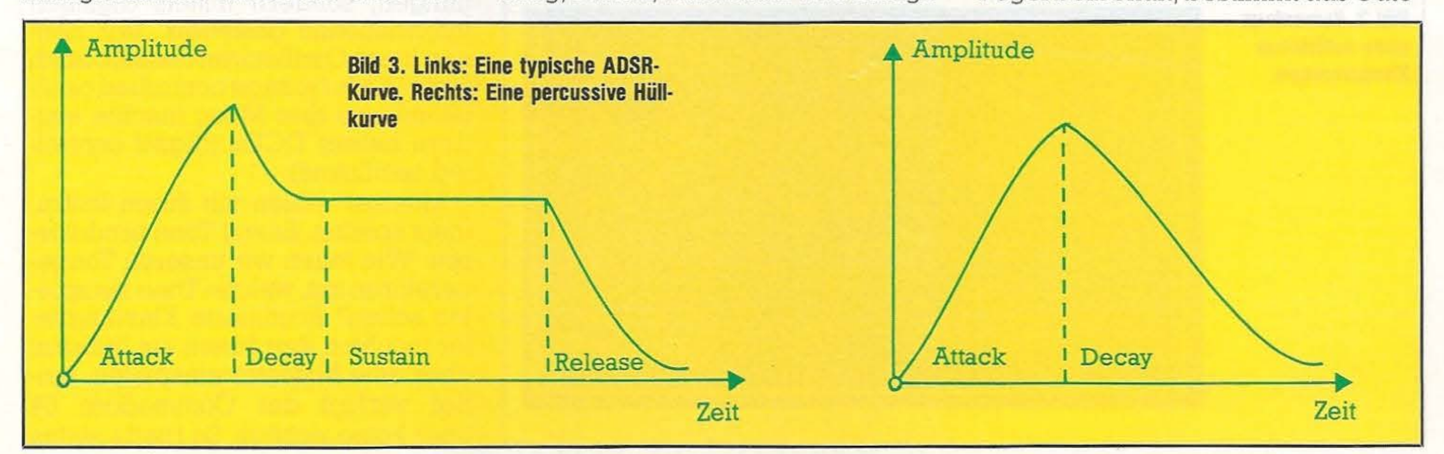


Bild 3. Links: Eine typische ADSR-Kurve. Rechts: Eine percussive Hüllkurve

Signal. Solange unser Gate gleich 1 ist, bleibt der Ton auf dem durch den eingestellten Sustain-Pegel bestimmten Lautstärkeniveau. Wird das Gate-Signal gleich 0, geht der Ton in die letzte der vier Phasen, die Release-Phase über.

Die Release-Phase bestimmt, in welcher Zeit unser Ton vom Sustain-Pegel auf die Lautstärke Null abfällt. Sobald unser Gate gleich 0 wird, beginnt die Release-Phase. Natürliche Instrumente besitzen nur kurze Release-Phasen. Man denke zum Beispiel an ein Klavier. Läßt man die Tasten nach dem Anschlag wieder los, verklingt der Ton relativ schnell. Die Zeit des Tastendrucks entspricht der Zeit, die wir unser Gate-Signal auf dem Wert 1 halten, also eingeschaltet haben müssen. Manche Klänge durchlaufen nicht alle vier Phasen der Hüllkurve. Denken wir zum Beispiel an die Orgel. Der Orgelton erklingt fast augenblicklich nach dem Anschlagen einer Taste und verlischt ebenso schnell wieder. Der elektronisch erzeugte Klang benötigt im Vergleich zu einem natürlichen Instrument keine Zeit, um sich aufzubauen, und es schwingt auch kein Resonanzkörper nach. Um das zu realisieren, müßten wir Attack, Decay und Release gleich Null setzen, den Sustain-Pegel auf Maximum einstellen. Percussionsinstrumente wiederum verfügen nur über Attack und Decay. Trifft der Schlegel die Pauke, baut sich der Ton fast sofort auf. Genau so schnell klingt er jedoch wieder ab. Man kann ihn nicht durch längeres Drücken auf das Fell in eine Sustain-Phase zwingen. Ein Beispiel für so eine percussive Hüllkurve sehen wir rechts in Bild 3.

Unser SID-Chip wird digital gesteuert

Durch Auswahl verschiedener Grundschwingungsformen können wir unsere Klänge schon in bestimmte Richtungen lenken. Wir können diese Grundklänge jedoch noch weiter verändern.

Jede Stereo-Anlage verfügt über einen Klangregler. Wie jeder weiß, kann man mit diesem die Musik hell oder dumpf klingen lassen. Elektronisch wird dies mit einem Filter realisiert. Solche Filter kann man mit Kaffee-Filtern vergleichen. Diese halten Kaffekörnchen bestimmter Größe zurück. Ab einer bestimmten Größe können sie nicht mehr durch die Filterporen schlüpfen. Unsere Elektronik-Filter tun auch nichts anderes. Nur daß sie nicht Kaffeepartikel bestimmter Größe aufhalten, sondern harmonische Oberschwingungen bestimmter Frequenz. Wie wir bereits wissen, verändert dies unseren Gesamtklang unter Umständen beträchtlich. Nimmt man zum Beispiel alle Oberschwingungen höherer Frequenz weg, wird der Klang immer dumpfer. Nimmt man alle tieffrequenten Oberschwingungen weg, klingt der Ton heller. Auf diese Weise können wir also unsere Grundwellenformen nochmals entscheidend im Klang verändern.

Filter bestimmen, ob unser Ton hell oder dumpf klingt

Unser SID stellt zunächst drei verschiedene Grundfiltertypen zur Verfügung. Der Hochpaßfilter läßt alle hochfrequenten Oberschwingung passieren. Die Tiefen schneidet er ab. Leitet man einen Klang hindurch, klingt er am Schluß heller als zuvor. Der Tiefpaßfilter stellt das Gegenteil unseres Hochpasses dar. Er beschneidet die hohen Klanganteile, läßt tiefe ungehindert passieren. Die Klänge klingen dann dumpfer als vorher. Der dritte Filtertyp ist der Bandpaßfilter. Er läßt ein bestimmtes Frequenzband passieren, schneidet alle darüber- und darunterliegenden Frequenzanteile ab. Die Klänge werden dadurch flach.

Bei allen drei Filtertypen kann man die jeweilige Filterfrequenz bei der sie wirken sollen, genau einstellen.

Neben diesen drei Hauptfiltertypen, kann unser SID durch Mischung dieser drei, noch weitere Mischfiltertypen erzeugen.

Neben der Filterfrequenz existiert noch eine weitere wichtige klangbestimmende Größe, die Filterresonanz. Eine Gitarre klingt völlig anders als eine Geige, auch wenn man bei beiden die selben Töne spielt. Dies rührt von der unterschiedlichen mechanischen Bauweise der Resonanzkörper der beiden Instrumente her. Eine Elektrogitarre ohne solchen klingt ohne angeschlossenen Verstärker fast gar nicht. Der Resonanzkörper ist für den Klang verantwortlich. Er dient bei unseren Instrumenten gewissermaßen als Klangverstärker. Die Bauweise entscheidet hierbei, welche Frequenzen besonders verstärkt werden, welche weniger. Dieses Phänomen nennt man Resonanz.

Bei unserem elektronischen Filter kann natürlich kein Gehäuse mitschwingen. Die Resonanz wird elektronisch erzeugt. Welche Frequenzen dabei besonders angehoben werden sollen und welche nicht, können wir mit der sogenannten Filterresonanz einstellen. Auf diese Weise läßt sich das Resonanzverhalten natürlicher Instrumente simulieren.

Unser Sound-Chip besitzt nur einen Filter. Durch diesen müssen alle drei Tonoszillatoren geleitet werden. Sie lassen sich folglich nicht unabhängig voneinander mit diversen Filtereinstellungen versehen. Wir können aber für jeden Oszillator bestimmen, ob er den Filter durchlaufen soll oder nicht. Wir können die Oszillatoren, also auch um den Filter herum, direkt in den Verstärker leiten.

Ein weites Feld an Klangeffekten eröffnen Ringmodulation und Synchronisation von Oszillatoren. Ich möchte hier weniger auf die theoretischen Grundlagen dieser beiden Effekte eingehen. Nur so viel sei gesagt, in beiden Fällen wird ein Oszillatorsignal mit einem zweiten, von einem anderen Oszillator, moduliert.

Bei der Ringmodulation multipliziert man die Frequenzen der beteiligten Oszillatoren miteinander. Es entsteht dann eine neue Frequenz, die viele nicht harmonische Obertöne enthält. Im SID werden natürlich keine Frequenzen sondern digitale Zahlenwerte multipliziert. Auf diese Weise kann man vor allem metallische oder glockenähnliche Klänge synthetisieren.

Synchronisiert ein Oszillator einen anderen, so zwingt er diesen dazu, seine Kurvennulldurchgänge den eigenen anzupassen. Vor allem, wenn beide Oszillatoren unterschiedliche Frequenz aufweisen, ergeben sich hier interessante Klangeffekte.

So, dies war nun die graue Theorie, zumindest die erste Lektion auf dem steinigen Weg, dem SID-Chip Töne, Klänge und Musik zu entlocken. In einer weiteren Lektion werden wir lernen, mit welchen Befehlen wir die Parameter der Oszillatoren, Hüllkurvengeneratoren, des Filters und des Ringmodulators unseres SID-Chips ansprechen können.

(Richard Aicher/aa)

»Der Sumpf« Unser Beitrag über die Raubkopierer-Szene in der 64'er 6/84 hat weite Wellen geschlagen. Nachstehend bringen wir einen in mehrfacher Hinsicht bemerkenswerten Leserbrief zu diesem Thema — und unsere Antwort darauf.

Als Vorwegnahme meiner Meinung zu Ihrem Artikel »Der Sumpf« muß ich Ihnen sagen, daß ich selbst Raubkopierer bin. Ich bin 15 Jahre und mir voll und ganz bewußt, daß mein Tun illegal ist. Jedoch läßt sich von dieser »kleinen« Taschengeldaufbesserung (zirka 100 Mark pro Monat) ganz gut leben. Ich möchte aber Ihre These, »die Raubkopierer sind für die hohen Softwarepreise verantwortlich« widerlegen. Denn was war eher, die Software oder die Raubkopie? Es ist doch logisch, daß man ohne die teure Software, die urheberrechtlich geschützt ist, keine Raubkopie machen kann. Ich stimme Ihnen voll und ganz zu, daß es (auf gut deutsch gesagt) eine Schweinerei ist, daß Kinder genau dieselbe Strafe bekommen wie Profihacker. Ich will Ihnen für die wirklich gute Idee der billigen Kleinanzeigen keine Vorwürfe machen, sie ist nur ein weiterer guter Bestandteil Ihres sonst hervorragenden Heftes, aber meinen Sie nicht, daß Sie damit Raubkopierer etwas animieren? Meinen Brief unterzeichne ich mit

H. Acker

Es ist richtig, wenn Sie meinen, nicht die Raubkopierer allein wären für die hohen Software-Preise verantwortlich zu machen. Dazu kommt natürlich auch der oft enorme Entwicklungsaufwand, der in so einem Programm steckt. Nicht zuletzt wollen auch der Hersteller, die Distributoren und Händler einen kleinen Gewinn nach Hause tragen.

Es ist traurig aber wahr, daß die kleinen (sprich jungen) Raubkopierer mit demselben Strafmaß zu rechnen haben wie die großen. Die Firmen gehen in dieser Hinsicht aber nach dem Grundsatz: »Wehret den Anfängen« vor. Ein amerikanischer Datenschützer sprach einmal von der »fehlenden Moral« im Bereich der Software. Es ist in der Tat unter den Jugendlichen ein schwereres Vergehen, jemandem die Cola wegzutrinken, als einem relativ anonymen Entwickler das geistige Eigentum in Form eines Spiels oder Anwenderprogramms zu klauen. Sicher, die Materie, das Programm ist etwas reell nicht Greifbares; außer auf der Diskette (und das sollte man ja tunlichst vermeiden).

Die von Ihnen angesprochenen Kleinanzeigen in unserer Zeitschrift sind da nur ein Spiegel der Gesellschaft. Geplant waren sie, um dem Leser eine billige Möglichkeit zu bieten, Kontakte zu knüpfen, Erfahrungen auszutauschen, nicht mehr benötigte Hard- und Software zu verkaufen und vieles andere mehr. Nicht beabsichtigt war natürlich, den Raubkopierern ein preisgünstiges Forum für ihre illegalen Geschäftspraktiken zu sein. Wir sind um Abhilfe bemüht, aber wie läßt sich eine Anzeige über 100 selbstgeschriebene Programme von 100 raubkopierten unterscheiden, wenn dies nicht im Text erscheint. Außerdem gibt es in einigen Bereichen, wie zum Beispiel dem Lehrberuf, sogenannte »Public Domain Software«, die der Öffentlichkeit zugänglich gemacht wird.

Es ist nicht unsere Aufgabe und entspricht auch nicht unserem Selbstverständnis, der Vorreiter für die Industrie zu sein. Doch es ist unser Bestreben, ein in allen Belangen »sauberes« Magazin für den Computer-Fan zu machen. Dazu gehört es auch, rechtzeitig zu warnen, Mißstände aufzuzeigen und Hilfestellung zu leisten. Und diese Art der Taschengeldaufbesserung ist nicht in Ordnung. Die jungen Schüler und Studenten, denen es gelingt, den Software-Schutz fremder Programme zu knacken, eigene Kommentare und Veränderungen im Programm anzubringen, die verstehen doch etwas vom Programmieren. Diese Energien und dieses Know-how ließe sich doch wesentlich sinnvoller für die Entwicklung eigener, guter Programme einsetzen. Es gibt genug Programme auf dem Markt, die es nicht einmal wert sind, kopiert zu werden. Dagegen sollte man mit seinem ganzen Können angehen. Gerade in Deutschland sind wir in der Computertechnologie und der Softwareerstellung noch um Jahre hinter den USA, Großbritannien oder Japan. Es ist an der Zeit, denen zu zeigen, wo der Bartel den Most holt. Gute Software erstellen, zu einem fairen Preis, da liegt Eure Chance. (aa)

FEHLERSUCH

2. TEIL

Wie schon in der vorletzten Ausgabe des 64'er-Magazins angedeutet wurde, sind Eintipp-Fehler in DATA-Zeilen besonders tückisch. Und das hauptsächlich aus zwei Gründen. Erstens zeigt eine eventuelle Fehlermeldung nie auf den wirklichen Fehler, sondern in der Regel auf die Einlese- (READ-) Schleife dieser Daten. Zweitens, und das ist noch viel unangenehmer, kann es vorkommen, daß es gar keine Fehlermeldung gibt, sondern der Computer einfach aussteigt und »abstürzt«. Hoffentlich haben Sie das mühsam eingegebene Programm vor dem ersten RUN abgespeichert!

Gerade Anfänger haben jetzt Schwierigkeiten. Und deshalb möchte ich mich hauptsächlich an diese wenden.

Als erstes möchte ich Ihnen empfehlen, noch einmal das »Handbuch« des VC 20/C 64 in die Hände zu nehmen. Lesen Sie noch einmal durch, was dort über den READ/DATA-Befehl geschrieben steht. Auch ich bin diesem ominösen READ-Befehl anfangs möglichst aus dem Weg gegangen und stehe ihm jetzt noch manchmal mit Mißtrauen gegenüber. Und das liegt an den oben genannten zwei Punkten.

Oft sind nämlich diese DATA-Werte nichts anderes als codierte Maschinensprache. Und dann kann sogar eine einzige falsch abgetippte Ziffer zum absoluten Kollaps führen. Aber wenn es sich um Maschinensprache handelt, haben alle DATA-Werte einige gemeinsame Merkmale: Erstens sind sie nicht negativ, zweitens sind sie niemals größer als 256 und drittens sind es immer ganze Zahlen. Diese Merkmale können wir benutzen um eine kleine Prüfroutine zu entwickeln, die einen, aufgrund eines Eingabefehlers, falschen DATA-Wert sofort sichtbar macht, wenn er gegen diese Merkmale verstoßen sollte.

Dieses folgende kleine Programm sollten Sie vor oder hinter das zu überprüfende Programm

E IN BASICPROGRAMMEN

Immer wieder erreichen uns Anrufe von Lesern, die (besorgt) anfragen, ob in diesem oder jenem Listing Fehler bekannt sind. Meistens handelt es sich jedoch um Eintipp-Fehler, vom Leser selbst verursacht. Und meistens liegen die tückischen Fehler innerhalb von DATA-Zeilen. Was tun?

schreiben. Nach abgeschlossener Prüfung kann es ruhig wieder gelöscht werden. Hier das Programm:

```
50000 READ A$; PRINT "ZEILE "PEEK(64)*256+PEEK(63);A$
50010 POKE198,0: WAIT 198,1:GOTO 50000
```

Starten Sie das Programm in diesem Fall mit RUN 50000. Sie werden jetzt den ersten DATA-Wert am Bildschirm sehen, davor die Zeilennummer, in der dieser Wert steht. Wenn Sie dann eine Taste drücken , zum Beispiel die große Leertaste, wird der nächste Wert sichtbar, direkt darunter. Bei fortwährendem Drücken der Leertaste, rasen die Zahlen sehr schnell über den Bildschirm. Für den ersten Durchlauf sollten Sie das ruhig machen. Man sieht dann sofort, ob eine Zahl (unzulässiger Weise?) länger als drei Ziffern ist. Falls dies der Fall ist, können Sie sehr schnell im Originallisting nachschauen, ob der Wert in Ordnung ist. Falls der Fehler so jedoch nicht erkannt werden kann, bleibt nichts anderes übrig, als das Programm noch einmal zu starten und diesmal jeden einzelnen Wert mit dem Original zu vergleichen. Machen Sie sich aber keine Sorgen über die Fehlermeldung »OUT OF DATA ERROR« die unweigerlich am Ende auftaucht. Sie zeigt hier lediglich, daß alle DATAs angezeigt wurden.

Apropos Fehlermeldung. »OUT OF DATA ERROR« oder »ILLEGAL QUANTITY ERROR«. Wenn nach dem Starten des Hauptprogramms eine dieser Fehlermeldungen kommt, können Sie fast todsicher davon ausgehen, daß ein DATA-Wert falsch eingetippt wurde. Das »fast« nur deswegen, weil Sie eventuell die FOR-NEXT-Schleife, in der ein READ-Befehl steht, falsch eingegeben haben. Die Bedeutung dieser Fehlermeldungen sind im schon erwähnten Handbuch erklärt.

Prüfsummliste. Das sind die Prüfsummen des in dieser Ausgabe abgedruckten Mailbox-Programms.

PRUEFSUMMENLISTE
BLOCKGROESSE 30

ZEILE	ANZAHL	SUMME	KEIN POKE?
10030			
10040	30	3564	
10040			
10060	60	6829	
10090	90	10532	
10100			
10100			
10110	120	14114	
10140	150	16953	
10150			
10160	180	[illegible]	
10170			
10190	210	24477	
10210	240	27726	
10220			
10240	270	31445	
GESAMT	282	32970	

Listing. Diese Prüfroutine soll in Zukunft die Suche nach DATA-Fehlern erleichtern. Näheres im Text.

```
62000 REM-----------------------------
62010 REM PRUEFSUMMENLISTE
62020 REM-----------------------------
62030 BL$="          "
62040 INPUT"BLOCKGROESSE";BG
62050 OPEN1,4:CMD1
62060 PRINT"PRUEFSUMMENLISTE"
62070 PRINT"BLOCKGROESSE";BG
62080 GOSUB62200
62090 PRINT"ZEILE  ANZAHL   SUMME     KEI
N POKE?"
62100 GOSUB62200
62110 RESTORE
62120 READA$:A=VAL(A$)
62130 IFA$="*"THENGOSUB62200:RC$="":GOSU
B62210:PRINT"GESAMT" RC$:GOTO62300
62140 AN=AN+1:S=S+A:B=B+1
62160 IFA$<>MID$(STR$(A),2)THENPRINTPEEK
(64)*256+PEEK(63);SPC(20);A$
62170 IFA>256THENPRINT PEEK(64)*256+PEEK
(63) SPC(20);A$
62175 IFA<>INT(A)THENPRINT PEEK(64)*256+
PEEK(63) SPC(20);A$
62180 IFAN=BGTHENGOSUB62205  :PRINTRC$:A
N=0
62190 GOTO62120
62200 FORI=1TO10:PRINT"————";:NEXT:PRINT
:RETURN
62205 REM-----------------------------
62206 RC$=""
62207 Z$ =STR$(PEEK(64)*256+PEEK(63))
62208 RC$=RC$+LEFT$(BL$,6-LEN(Z$))+Z$
62210 B$=STR$(B):S$=STR$(S)
62220 RC$=RC$+LEFT$(BL$,6-LEN(B$))+B$
62240 RC$=RC$+LEFT$(BL$,10-LEN(S$))+S$
62250 RETURN
62260 DATA*
62300 PRINT#1:CLOSE1
READY.
```

Ich hoffe, daß Sie mit dieser kleinen Routine in Zukunft schneller diese lästigen Fehler finden. Um Ihnen aber die Fehlersuche noch etwas weiter zu erleichtern, habe ich ein weiteres Programm geschrieben. Wir werden in den kommenden Heften bei Programmen mit vielen DATA-Werten eine spezielle Prüfsummenliste mit abdrucken, die es gestattet, alle DATA-Werte blockweise zu überprüfen. Dazu müßen Sie dann jedesmal das hier abgedruckte Programm (siehe Listing) an das zu testende Programm anhängen (entweder mit MERGE oder durch Abtippen) und laufen lassen. Damit kann ein Tippfehler auf wenige DATAs eingegrenzt werden. Wenn Sie das nebenstehende Programm laufen lassen, sollten Sie am Bildschirm die gleichen Werte erhalten wie die dann abgedruckte Prüfsummenliste. Eine Abweichung kann auf einen Eingabefehler hinweisen (muß aber nicht). Die vier Spalten der Prüfsummenliste haben folgende Bedeutung:

Ganz links steht unter ZEILE die Zeilennummer des letzten DATA-Wertes des jeweiligen Blocks, daneben die Anzahl der bisher gelesenen DATAs, rechts davon die Summe aller bisher gelesenen Werte und ganz rechts eine mögliche Fehlerquelle, das heißt, hier wird ein Wert angezeigt, wenn er größer als 256 ist oder einer gebrochenen Zahl, aber auch, wenn er keine Zahl ist und Buchstaben enthält.

Die als Beispiel abgedruckte Prüfsummenliste sieht so aus, wenn Sie den DATA-Tester an das in diesem Heft veröffentlichte Mailbox-Programm angehängt haben. Es wurden hier jeweils 30 DATAs gelesen und deren Prüfsumme errechnet. Man sieht auch, daß sich zum Beispiel in Zeile 10030 (die erste Zahl in der Prüfsummenliste) ein Leerstring befindet. An dieser Stelle werden Sie im Mailbox-Programm zwei Kommata finden. Das gleiche gilt für die Zeilen 10040,10100,10150,10170 und 10220. Insgesamt werden 282 DATAs gelesen. Die Gesamtsumme beträgt 32970. Die Spalte unter »TEXT« bleibt leer, weil in den DATAs weder Text noch negative Zahlen vorkommen.

Ich hoffe, daß Sie mit diesen Mitteln mögliche Eingabefehler schneller finden und somit nicht vorschnell Frust aufkommt. Aber einen Fehler werden Sie oft vergebens suchen: Das ist der, den unser Fehlerteufel hinzaubert. Aber dem werden wir in der Redaktion schon das Leben schwer machen (umgekehrt allerdings genauso). (gk)

SPRING VOGEL'S



Bild 5. Szene fünf ist nicht unlösbar, man braucht aber Ausdauer.

Bild 3. Die dritte Spielszene zeigt bereits neue Elemente wie Rutschbahn und Förderbänder

Bild 1. So sieht das Titelbild mit den Wahlmöglichkeiten und der Funktionstastenbeschreibung aus

Kennen Sie Jumpman, Miner 2049 oder Mister Robot and his Factory? Dann haben Sie sich sicherlich geärgert, daß Sie nie die letzten Bilder erreicht haben. Oder Sie haben sich über die eintönigen Spielszenen geärgert. Mit dem Listing des Monats wäre das nicht passiert. Spring-Vogel ist eine Kombination aus den drei Spielen. Eines kann jedoch nicht mehr passieren! Wird eines der Bilder langweilig, dann machen Sie sich einfach ein neues.

Tatsächlich kann man Spring-Vogel fast als Spielgenerator bezeichnen. Mit Spring-Vogel gelingt es Ihnen, sich jedes beliebige Spielfeld aufzubauen. Das klingt sehr vielversprechend, und das ist es auch.

Spring-Vogel ist ein Vertreter der Jump and Run-Kategorie. Damit lassen sich also keine Schieß- oder Abenteuerspiele erzeugen, aber innerhalb der Spring- und Laufgruppe bleibt kein Wunsch offen. Worum handelt es sich bei Spring-Vogel nun eigentlich? Zunächst die Story.

Ein heftiger Sturm hat einen Vogel — unseren Helden — mitsamt seinen Eiern aus dem Nest geweht. Durch den harten Aufprall auf die Erde hat er sich seine Flügel gebrochen. Er kann also nicht mehr richtig fliegen, sondern nur noch auf dem Boden laufen, hüpfen und springen. Die Aufgaben des Vogels (die genaue Klassifizierung bleibt Ihnen überlassen) besteht nun darin, in einem Labyrinth aus Aufzügen, Transportbändern, Seilen, Einbahnstraßen, Trampolinen, Rutschbahnen, Gummiwänden, magischen Flügeln, gemeinen Vogelfallen und mißgestimmten Monstern alle Eier wieder einzusammeln. Gelingt es dem Vogel, die Eier in einer Spielszene aufzunehmen, muß er mit der nächsten Überraschung fertig werden; es waren nicht alle. Das nächste vom Winde verwehte Bild erwartet ihn mit weiteren schwierigen Aufgaben.

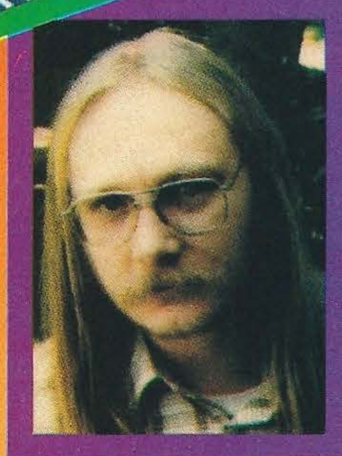
Die Anzahl der Torturen für unseren leidgeprüften Helden bestimmen Sie selbst. Doch Vorsicht, unser Vogel verfügt nicht wie eine Katze über sieben, sondern »nur« über sechs Leben. Und diese sechs Leben sind schnell, durch Kontakt mit den Monstern oder Verzehrern einer roten Tollkirsche, ausgehaucht.

PRING

Bild 2. Die erste Spielszene ist relativ leicht zu bewältigen

Bild 4. Szene vier. Nicht einfach und dadurch reizvoll

Der



Gewinner

Sein Ärger brachte Matthias Törk 2000 Mark ein.

Ich bin am 18.4.1955 geboren und studiere derzeit Germanistik, Philosophie und Informatik. Mein erster Computer war ein VC 20, der dann Anfang 1983 durch einen C 64 ersetzt wurde.

Für die Entwicklung des Programms »Spring-Vogel« ist neben der verlockenden Aussicht auf 2000 Mark vor allem der Ärger über Spiele wie »Jumpman Jnr.« oder »Miner 2049er« verantwortlich, bei denen ich die letzten Bilder nie zu sehen bekam. Nach kleinen Änderungen an diesen Programmen, die die Helden mit Unsterblichkeit versorgten, war zwar das Ende einfach zu erreichen, aber leider wurden die Spiele dadurch auch schnell langweilig. Deshalb wollte ich ein ähnliches Spielprogramm entwickeln, daß diese Nachteile vermeidet.

Zunächst sollte es möglich sein, jedes einzelne Bild extra anzuwählen und vor allem sollten sich einfach neue Bilder erstellen lassen sobald die alten langweilig werden. Durch konsequente Parametrisierung und eine Spielroutine, die in der Art eines Interpreters die Bilder und Tabellen abarbeitet, entwickelte sich aus dieser Idee schließlich ein regelrechter »Generator« für solche »Jump an Run«-Spiele. Das Ergebnis liegt nun nach rund einem Monat Entwicklungszeit in Form von »Spring-Vogel« vor, einem Programm, mit dem Sie sicher viel Spaß haben werden. (Matthias Törk)

Der Sturz über mehr als vier Etagen ist ebenfalls für den flugunfähigen Vogel lebensbedrohend. Es sei denn, er findet die von seinem Tierschützer verstreuten »magischen Flügel«. Ausgestattet mit deren zauberhaften Fähigkeiten kann er in einem ökologischen Flecken beliebig umherfliegen. Durch Betätigen des Feuerknopfes wird er wieder in seinen behinderten Zustand zurückversetzt, erlangte allerdings vorher die Fähigkeit an unzugänglichen Stellen landen zu können. Diese magischen Flügel gestatten es, Bilder so aufzubauen, daß nur der taktisch gezielte Einsatz dieser Flughilfe es ermöglicht, ein Bild vollständig abzuräumen. Dabei kann es vorkommen, daß sich der Vogel in eine völlig aussichtslose Situation manövriert, und in einer Sackgasse landet. Auch hier bietet das Programm einen Ausweg an, die Funktionstasten. Diese können jederzeit (außer während der Flugphase) benutzt werden.

Die genaue Bedeutung können Sie der Beschreibung zum Listing entnehmen. Um zu sehen, welcher Vogel von welchem Spieler der bessere ist, gibt es auch hier eine Punktezählung. Bei den bereits mitgelieferten Bildern ist diese in Zehnerschritten ab dem ersten Bild gestaffelt.

Entscheidungskriterien

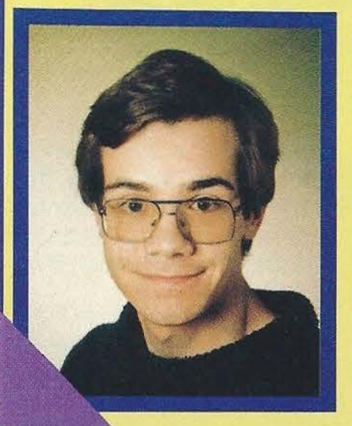
Weshalb wurde dieses Programm zum Listing des Monats auserkoren? Das Hauptkriterium war die Realisierung eines Spielegenerators. Mit dem Editiermodus von Spring-Vogel lassen sich nahezu beliebig viele Bilder konstruieren. Wird ein Bild langweilig, so kann man ohne großen Aufwand ein neues mit noch größeren Schwierigkeiten entwickeln (versuchen Sie aber erst einmal, die beiden bereits existierenden Bilder 5 und 6 zu bewältigen). Der Phantasie sind keine Grenzen gesetzt.

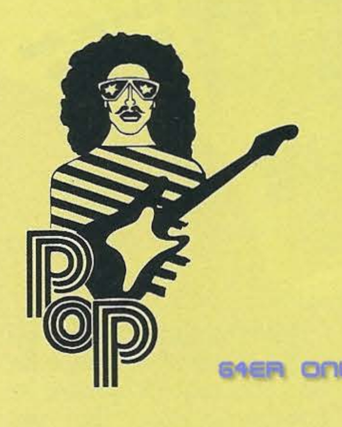
Warum alleine spielen?

Vorstellbar ist ein Wettbewerb unter Freunden, wer das lustigste, das interessanteste, das schwierigste oder das raffinierteste Bild entwirft. Der Programmieraufwand hält sich für ein solch komplexes Programm in relativ bescheidenen Grenzen. Die vorgegebenen Bausteine bieten eine derartige Vielfalt an verschiedenen Bildern, daß jeder nach seinem eigenen Geschmack entwerfen kann. Insgesamt ein Programm, das durch Witz und Programmierkunst überzeugt.

(aa/Matthias Törk)

Die Musik macht de





Der Orgelbauer

Wohlklingende Münze erhielt Christian Gebauer für sein Musikprogramm. Der Autor stellt sich selbst kurz vor.

Ich wurde am 21. Oktober 1965 in Bad Nauheim geboren. Nach der Grundschule besuchte ich das Ernst-Ludwig-Gymnasium, an dem ich gerade die 12. Klasse absolviert habe und mich auf das Abitur vorbereite.

Meine Kombination der Leistungskurse ist Chemie und Mathematik, beides Fächer, die mich sehr interessieren. In diesem Jahr belegte ich einen EDV-Kurs, der von der Schule angeboten wurde.

Zur Computerei bin ich durch einen Bekannten gekommen, der mir im Herbst 1982 für mehrere Wochen einen Computer geliehen hat und mich in Basic einweihte. Meine Kenntnisse dieser Sprache vertiefte ich dann im Selbststudium.

Im darauffolgenden Frühjahr kauften mir meine Eltern

So fing es an

einen Commodore 64 mit Floppy 1541, ein halbes Jahr später kam dann noch ein Seikosha-Drucker GP-100VC hinzu.

Das Programm »Orgel« begann ich im Sommer 1983 zu schreiben, um die hervorragenden Toneigenschaften des Gerätes auszunutzen. Diese Version ist das Ergebnis eines halben Jahres Erweiterungs- und Verbesserungsarbeit.

(Christian Gebauer)

```
0 GOTO 100
1 REM"                ----
2 REM"     |        --------          |
3 REM"     ||    ---------------      ||
4 REM"     |||    ELEKTRONENORGEL     |||
5 REM"     ||    ---------------      ||
6 REM"     |        --------          |
7 REM"                ----
9 :
10 REM           CHRISTIAN GEBAUER
11 REM           6350 BAD NAUHEIM
12 REM              APRIL 1983
13 REM         VERBESSERT IM JULI '83
15 :
19 REM
20 REM           --- VARIABLEN ---
25 REM
30 REM IZ        ZAHL DER INSTRUMENTE
31 REM MZ         ''   ''  TOENE
32 REM TZ         ''   ''  TONTASTEN
33 REM
34 REM I$()      INSTRUMENTENNAME
35 REM IB$(,)    SYMBOLFRAGMENT DES INSTR.
36 REM I1()      PARAMETER FUER DIE
37 REM  --        KLANGEIGENSCHAFTEN
38 REM I5()       DES INSTRUMENTES
                  ENTSPR.: WF,AS,HA,TL,TH
39 REM IA        AKTUELLES INSTRUMENT
40 REM IB        VORIGES    ''
41 REM IH        GRUNDEISTELLUNG HALL-EFF.
42 REM IO         ''                OKTAVE
43 REM
44 REM M1%()     TONPARAMETER HIGH
45 REM M2%()      ''           LOW
46 REM N1        ERGEBNIS TASTE-TON LOW
47 REM N2         ''        ''      HIGH
48 REM
49 REM T%()      TASTEN-TON-ZUORDNUNG
50 REM
51 REM SID-ADRESSEN:
52 REM SI()      GRUNDADRESSE
53 REM FL()      FREQUENZ LOW
54 REM FH()      FREQUENZ HIGH
55 REM TL()      TASTVERHAELTNIS LOW
56 REM TH()      TASTVERHAELTNIS HIGH
57 REM WF()      WELLENFORM
58 REM AS()      ANSCHLAG - ABSCHWELLEN
59 REM HA()      HALTEN - AUSKLINGEN
60 REM
61 REM O         OKTAVE (1 ENSPR. 28)
```

Listing »Elektronenorgel«

Wie musikalisch der C 64 sein kann, ist den meisten bekannt, wie man die Musik macht, aber den wenigsten. Dieses Programm, als Anwendung des Monats, zeigt die Fähigkeiten Ihres Computers.





Elf verschiedene Instrumente sind in diesem Programm schon verwirklicht. Sie können zwischen Piano, Röhrengong, Metallophon, Xylophon, Glocke, Glasorgel, Violine, Flöte, Panflöte, Klarinette und Harfe auswählen. Das Programm läßt sich jedoch noch erweitern. Ihrer Phantasie sind keine Grenzen gesetzt.

Empfehlenswert ist ein Verstärker oder eine Stereoanlage. Das Programmlisting ist weitgehend selbstdokumentierend, daher nur einige Hinweise zur Bedienung:

Oben rechts wird eine Klaviatur dargestellt, die der augenblicklichen Tastenbelegung entspricht. Gespielt wird durch Drücken der jeweiligen Taste.

Die Instrumente werden durch geshiftete Buchstaben gewählt.

Besondere Funktionen:

— Halleffekt:
Alle drei Tongeneratoren werden abwechselnd benutzt, so daß immer zwei Töne nachklingen. Symbol: ⟨⟨⟨0⟩⟩⟩ ganz unten.

— Oktavenwahl:
Oktaven wählbar von Subcontraoktave (SCO) bis dreigestrichene Oktave (0'''). Die Subcontraoktave ist nur durch Einzeltonverschiebung zu erreichen. Das Pfundzeichen setzt Instrumente in die Grundoktave zurück und die Einzeltonverschiebung 0. Symbol: unter der Klaviatur.

— Einzeltonverschiebung:
Die Tastaturbelegung kann in Einzeltonschritten über den gesamten Tonbereich verschoben werden. Symbol: Klaviatur.

— Akkorde:
Zweiklang, wählbar bis None. Apostroph setzt Akkorde auf 0. Symbol: >?< links neben Halleffekt.

— Lautstärke:
wählbar von 0 bis 15.
Symbol: 0000000000...unten.

Von der Bildschirmaufteilung her kann noch ein Instrument zusätzlich entworfen werden. Dazu muß man die Variable IZ in 150 gleich 11 setzen. Nun kann man ab Zeile 620 das neue Instrument entwerfen.

Format:
Name, Wellenform, Anschlag-Abschwellen, Halten-Ausklingen. Tastverhältnis low, Tastverhältnis high, Hall (3 = ja, 1 = nein), Grundoktave (Contraoktave = 0), Bild (10 x 7).

Soll mehr als ein neues Instrument entworfen werden, muß man die Bildschirmgestaltung ändern.

(Christian Gebauer/rg)



```
62 REM V      VERSCHIEBUNG
63 REM W      VERSCHIEBUNG DES MANUALS
64 REM AK     AKKORD
65 REM H      HALLEFFEKT: 1=AUS  3=EIN
66 REM G      AKTUELLER TONGENERATOR
67 REM GV     VORIGER TONGENERATOR
68 REM D      GRUNDTONKONSTANTE (D+T)
69 REM L      LAUTSTAERKE
70 REM
71 REM A$     EINGABESTRING
72 REM A      NUM. FUER A$
73 REM P      FLAG FUER GEN. AUSSCHALT.
74 REM
75 REM MD$()  MANUALDARSTELLUNG
76 REM MB$()  BUCHSTABEN DES MANUALS
77 REM
80 REM I,J,S$ HILFSVARIABLEN
90 :
100 REM"          ─────────────
101 REM"         ╭─────────────╮
102 REM"         │  VORSPANN   │
103 REM"         ╰─────────────╯
104 REM"          ─────────────
110 :
120 PRINT"[CLR]";CHR$(142);CHR$(8):POKE 650,
0:POKE 53280,14:POKE 53281,6
130 PRINT"[RVS]"
140 :
150 IZ=10:MZ=111:TZ=33:L=10
160 DIM M1%(MZ),M2%(MZ)
170 DIM I$(IZ),IB$(IZ,7),I1(IZ),I2(IZ),I
3(IZ),I4(IZ),I5(IZ),IH(IZ),IO(IZ)
180 DIM T%(TZ),MD$(40),MB$(40)
190 :
200 REM     ─────────────────────
201 REM     --- INSTRUMENTE ---
202 REM     ─────────────────────
210 :
220 FOR I=0 TO IZ
240 READ I$(I),I1(I),I2(I),I3(I),I4(I),I
5(I),IH(I),IO:IO(I)=14*IO
250 FOR J=1 TO 7
260 READ IB$(I,J)
270 NEXT J
280 NEXT I
290 :
300 DATA         PIANO
305 DATA 65,9,0,0,202
307 DATA 3,3
310 DATA "      /        "
311 DATA "     /_        "
312 DATA "    // \___    "
313 DATA "   //______\   "
314 DATA "  |_________|  "
315 DATA "   !       !   "
316 DATA "   ▪       ▪   "
320 :
330 DATA     ROEHRENGONG
335 DATA 65,11,0,0,200
337 DATA 3,3
340 DATA "|| || || || ||"
341 DATA "|| || || || ||"
342 DATA "|| || || || ||"
343 DATA "|| || |● ||   "
344 DATA "|| || ||\     "
345 DATA "|| ||    \    "
346 DATA "||          \ "
350 :
360 DATA     METALLOPHON
365 DATA 17,10,0,0,0
367 DATA 3,4
370 DATA "  █▄          "
371 DATA "  ███▄        "
372 DATA "  █████       "
373 DATA "  ████▀●      "
374 DATA "  ██▀  /      "
375 DATA "  ▀──/─●      "
376 DATA "     /        "
380 :
390 DATA       XYLOPHON
395 DATA 17,5,0,0,0
397 DATA 3,4
400 DATA "  ╭          "
401 DATA " || ╭        "
402 DATA " |||| ╭      "
403 DATA " |||||| /    "
404 DATA " ||||╰/      "
405 DATA " || ╰●─/──   "
406 DATA " ╰     ●     "
410 :
420 DATA        GLOCKE
425 DATA 17,13,0,0,0
427 DATA 3,4
430 DATA "     ╭─╮      "
431 DATA "     | |      "
432 DATA "     | |      "
433 DATA "    /___\     "
434 DATA "   |/  ●\|    "
435 DATA "    \___/     "
436 DATA "              "
440 :
450 DATA       GLASORGEL
455 DATA 17,170,0,0,0
457 DATA 3,4
460 DATA "    ____      "
461 DATA "   (____)     "
462 DATA "    \  /      "
463 DATA "     |        "
464 DATA "     |        "
465 DATA "    (_)       "
466 DATA "              "
470 :
480 DATA      VIOLINE
485 DATA 65,170,202,250,0
487 DATA 1,3
490 DATA "      ╭       "
491 DATA " \    ||      "
492 DATA "   \  ||      "
493 DATA "    / || \    "
494 DATA "    >)(<      "
495 DATA "    | || |    "
496 DATA "    \____/\   "
500 :
510 DATA       TROETE
515 DATA 33,90,228,0,0
517 DATA 1,3
520 DATA "              "
521 DATA "           /  "
522 DATA "-.___▪▪▪_‾    "
523 DATA "-'‾/‾‾‾‾‾‾_   "
524 DATA "   \___/  \   "
525 DATA "              "
526 DATA "              "
530 :
540 DATA     PANFLOETE
545 DATA 17,102,197,0,0
547 DATA 1,4
550 DATA "              "
```

Listing des Programms »Elektronenorgel« (Fortsetzung)

```
551 DATA "  ███████ "
552 DATA "  █████"" "
553 DATA "  ███"    "
554 DATA "  ██      "
555 DATA "  █       "
556 DATA "          "
560 :
570 DATA   KLARINETTE
575 DATA 65,105,197,0,100
577 DATA 1,2
580 DATA "  \       "
581 DATA " \O\      "
582 DATA "  \O\     "
583 DATA "   \O\    "
584 DATA "    \/‾\  "
585 DATA "     |  | "
586 DATA "     \_/  "
590 :
600 DATA     HARFE
605 DATA 17,44,0,0,6
607 DATA 3,3
610 DATA " /‾‾_‾‾‾‾‾_"
611 DATA "|\\\\\\\\\//"
612 DATA " |\\\\\\// "
613 DATA "  | \\\// "
614 DATA "   | \//  "
615 DATA "   | //   "
616 DATA "    //    "
900 :
1000 REM      --------------------
1001 REM      --- TONPARAMETER ---
1002 REM      --------------------
1010 :
1020 FOR I=1 TO MZ
1030 READ M2%(I),M1%(I)
1040 NEXT I
1050 :
1060 DATA 1,4,,
1065 :
1070 DATA 1,22,1,39,1,57,1,75,1,95,,,1,1
16,1,138,1,161,1,186,1,212
1080 DATA 1,240,2,14,,
1085 :
1090 DATA 2,45,2,78,2,113,2,150,2,190,,,
2,231,3,20,3,66,3,116,3,169
1100 DATA 3,224,4,27,,
1105 :
1110 DATA 4,90,4,156,4,226,5,45,5,123,,,
5,207,6,39,6,133,6,232,7,81
1120 DATA 7,193,8,55,,
1125 :
1130 DATA 8,180,9,56,9,196,10,89,10,247,
,,11,158,12,78,13,10,13,208
1140 DATA 14,162,15,129,16,109,,
1145 :
1150 DATA 17,103,18,112,19,137,20,178,21
,237,,,23,59,24,157,26,20,27,160
1160 DATA 29,69,31,3,32,219,,
1165 :
1170 DATA 34,207,36,225,39,18,41,101,43,
219,,,46,118,49,58,52,39,55,65
1180 DATA 58,138,62,5,65,181,,
1185 :
1190 DATA 69,157,73,193,78,36,82,201,87,
182,,,92,237,98,115,104,78,110,130
1200 DATA 117,20,124,10,131,106,,
1205 :
1210 DATA 139,59,147,130,156,72,165,147,
175,107,,,185,218,196,231
```

```
1220 DATA 208,156,221,4,234,40
1300 :
1500 REM     --------------------
1501 REM     --- DATA FUER TASTEN- ---
1502 REM        TON - ZUORDNUNG ---
1503 REM     --------------------
1510 :
1520 FOR I=0 TO 33
1530 READ T%(I)
1540 NEXT I
1560 DATA 23,20,22,24,,21,2,,,6,5,8,10
,12,15,14,16,18,,
1570 DATA 17,19,1,7,4,9,13,,3,,11,
1590 :
2000 REM     --------------------
2001 REM     --- MANUALDATA ---
2002 REM     --------------------
2010 :
2020 FOR I=1 TO 14
2030 READ MD$(I)
2040 NEXT
2050 FOR I=15 TO 40
2060 MD$(I)=MD$(I-14)
2070 NEXT
2080 :
2100 DATA " ","▚ █"," ","▚ █","▚|█","▚▚
▚█"," ","▚ █"," ","▚ █","▚|█","▚ █"
2110 DATA " ","▚ █"
2190 :
2200 FOR I=0 TO 24
2210 READ MB$(I)
2220 NEXT
2230 :
2240 DATA " ",Q,A,W,S,E,D,R,F,T,G,Y,H,U,
J,I,K,O,L,P,":",@,;,*,=
2290 :
3000 REM     --------------------
3001 REM     --- ADRESSEN DES SID ---
3002 REM     --------------------
3010 :
3020 SI(1)=54272
3030 SI(2)=SI(1)+7
3040 SI(3)=SI(1)+14
3050 FOR I=1 TO 3
3060 FL(I)=SI(I)
3070 FH(I)=SI(I)+1
3080 TL(I)=SI(I)+2
3090 TH(I)=SI(I)+3
3100 WF(I)=SI(I)+4
3110 AS(I)=SI(I)+5
3120 HA(I)=SI(I)+6
3130 NEXT I
3200 :
4000 REM     --------------------
4001 REM     --- GESTALTUNG DES ---
4002 REM     ---  BILDSCHIRMS  ---
4003 REM     --------------------
4010 :
4150 V=0:AK=0:G=1:IA=0
4200 :
4210 PRINT"..."
4250 :
4300 PRINT" INSTRUMENTE           SONSTIGES
":PRINT" (SHIFT)@":S$="*"
4310 FOR I=0 TO IZ
4320 IF S$<>"" THEN READ S$
4330 PRINT" "CHR$(I+65);TAB(3);I$(I);TAB
(20);S$
4340 NEXT I
```

Listing des Programms »Elektronenorgel« (Fortsetzung)

```
4350 DATA " /  HALL-EFFEKT"
4360 DATA "+\- OKTAVEN"
4370 DATA "< > TONVERSCHIEBUNG"
4380 DATA "(') AKKORDE"
4390 DATA "HOM LAUTSTAERKE +"
4400 DATA "DEL LAUTSTAERKE -",
4500 :
4700 GOSUB 10100
4710 GOSUB 11100
4720 GOSUB 12100
4730 GOSUB 13100
4740 GOSUB 14100
4750 GOSUB 15000
4760 GOSUB 16040
4770 :
4780 GOTO 5000
4790 :
4800 REM  -------------------
4801 REM  --- DARSTELLUNG ---
4802 REM  --- DES MANUALS ---
4803 REM  -------------------
4810 :
4820 PRINT"▒▒";
4825 W=V+1:IF W<0 THEN W=W+14
4830 FOR X=1 TO 2:PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒";
4835 FOR I=0 TO 23
4840 PRINT MD$(I+W);
4850 NEXT:PRINT:NEXT
4860 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒";
4870 FOR I=0 TO 23
4880 IF MD$(I+W)=CHR$(32) THEN PRINT MB$
(I+1);:GOTO 4900
4890 PRINT MD$(I+W);
4900 NEXT
4910 PRINT:PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒";
4920 FOR I=2 TO 24 STEP 2
4925 PRINT"|";
4930 IF MD$(I+W-1)="▒▒ ▒▒" THEN PRINT"▒
▒";:GOTO 4940
4935 PRINT" ";
4940 NEXT:PRINT"▒"
4950 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒";
4960 FOR I=2 TO 24 STEP 2
4965 PRINT"|";
4970 IF MD$(I+W-1)="▒▒ ▒▒" THEN PRINT"▒"
MB$(I)"▒";:GOTO 4980
4975 PRINT MB$(I);
4980 NEXT:PRINT"▒"
4990 RETURN
4999 :
5000 REM"
5001 REM" |┌─────────────────────┐
5002 REM" || LAUFENDES PROGRAMM ||
5003 REM" |└─────────────────────┘
5004 REM"
5010 :
5020 GET A$
5030 IF I3(IA)=0 THEN 5040
5035 IF PEEK(203)=64 AND P<>0 THEN FOR I
=1 TO 3:POKE WF(I),I1(IA)-1:NEXT:P=0
5040 IF A$="" THEN 5020
5050 :
5100 IF A$>="@" AND A$<="Z" THEN GOSUB 2
0000:GOTO 5020
5110 IF A$="*" OR A$=";" OR A$=":" OR A$
="=" THEN GOSUB 20000:GOTO 5020
5120 IF A$>="♠" AND A$<=CHR$(193+IZ) THE
N GOSUB 10000:GOTO 5020
5130 IF A$="+" OR A$="\" OR A$="-" THEN
GOSUB 11000:GOTO 5020
5140 IF A$="<" OR A$=">" THEN GOSUB 1200
0:GOTO 5020
5150 IF A$="(" OR A$=")" OR A$="'" THEN
GOSUB 13000:GOTO 5020
5160 IF A$="/" THEN GOSUB 14000:GOTO 502
0
5170 IF A$=CHR$(32) THEN GOSUB 15000:GOT
O 5020
5180 IF A$="▒" OR A$=CHR$(20) THEN GOSUB
 16000:GOTO 5020
5900 :
6000 GOTO 5020
7000 :
8000 REM"  ┌──────────────────────────┐
8001 REM"  |                          |
8002 REM"  | KLANG- UND TASTROUTINEN  |
8003 REM"  |                          |
8004 REM"  └──────────────────────────┘
9000 :
10000 REM  ----------------------
10001 REM  --- INSTRUMENTWAHL ---
10002 REM  ----------------------
10010 :
10020 IB=IA
10030 IA=ASC(A$)-193
10040 :
10100 PRINT"▒"
10110 FOR I=1 TO 7
10120 PRINT "  "IB$(IA,I)
10130 NEXT I
10140 :
10200 FOR I=1 TO 3
10210 POKE AS(I),I2(IA)
10220 POKE HA(I),I3(IA)
10230 POKE TL(I),I4(IA)
10240 POKE TH(I),I5(IA)
10250 NEXT I
10300 :
10310 H=IH(IA)
10320 O=IO(IA)
10330 GOSUB 14100
10340 GOSUB 11060
10390 :
10400 PRINT"▒"
10410 FOR I=0 TO IB+10
10420 PRINT
10430 NEXT I
10440 PRINT"▒▒▒"I$(IB)"▒"
10450 :
10460 FOR I=0 TO IA+10
10470 PRINT
10480 NEXT I
10490 PRINT"▒▒▒▒"I$(IA)"▒▒"
10500 :
10800 RETURN
10900 :
11000 REM  -----------------------
11001 REM  --- AKTUELLE OKTAVE ---
11002 REM  -----------------------
11010 :
11020 IF A$="\" THEN O=IO(IA):IF V<>0 TH
EN V=0:GOSUB 12100
11030 IF A$="+" THEN O=O+14
11050 IF A$="-" THEN O=O-14
11060 D=O+V+AK+11
11070 IF D<0 THEN O=O+14:GOTO 11060
11080 IF D>MZ-24 THEN O=O-14:GOTO 11060
11090 :
```

Listing des Programms »Elektronenorgel« (Fortsetzung)

```
11100 OK=INT((O+7)/14)
11110 PRINT"▒"
11120 PRINT"▒▒▒▒▒▒";TAB(13);
11125 IF OK=-1THEN PRINT"SCO ":RETURN
11130 IF OK=0 THEN PRINT"CO  ":RETURN
11140 IF OK=1 THEN PRINT"GO  ":RETURN
11150 IF OK=2 THEN PRINT"KO  ":RETURN
11160 PRINT"O";:FOR I=3 TO OK
11170 PRINT"'";
11180 NEXTI
11190 PRINT"   "
11300 :
11800 RETURN
11900 :
12000 REM  --------------------
12001 REM  --- VERSCHIEBUNG ---
12002 REM  --------------------
12010 :
12020 IF A$="<" THEN V=V-2
12030 IF A$=">" THEN V=V+2
12050 D=O+V+AK+11
12055 IF D>MZ-24 THEN V=V-2:GOTO 12050
12060 IF D<0 THEN V=V+2:GOTO 12050
12070 IF V>6 THEN V=-6:O=O+14:GOSUB 1110
0
12080 IF V<-6 THEN V=6:O=O-14:GOSUB 1110
0
12090 :
12100 GOSUB 4800
12130 :
12800 RETURN
12900 :
13000 REM  --------------
13001 REM  --- AKKORDE
13002 REM  --------------
13010 :
13020 IF A$="'" THEN AK=0
13030 IF A$="(" THEN AK=AK-2
13040 IF A$=")" THEN AK=AK+2
13050 D=O+V+AK+11
13055 IF D<0 THEN AK=AK+2:GOTO 13050
13060 IF D>MZ-24 THEN AK=AK-2:GOTO13050
13070 IF ABS(AK)>16 THEN AK=16*SGN(AK):G
OTO 13050
13090 :
13100 PRINT"▒"
13110 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒";TAB(
20);
13120 PRINT" )";AK/2+SGN(AK);"( "
13130 :
13800 RETURN
13900 :
14000 REM  ---------------
14001 REM  --- HALL-EFFEKT
14002 REM  ---------------
14010 :
14020 IF H=1 THEN H=3:GOTO 14100
14030 H=1
14090 :
14100 IF H=3 THEN H$=")))●((("
14110 IF H=1 THEN H$="       ":G=1
14120 :
14150 PRINT"▒"
14160 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒";TAB(
30);
14170 PRINT H$
14190 :
14800 RETURN
14900 :
15000 REM  ---------------------
15001 REM  --- TONGENERATORENSTOP
15002 REM  ---------------------
15010 :
15020 FOR I=1 TO 3
15030 POKE WF(I),0
15040 NEXT I
15050 :
15800 RETURN
15900 :
16000 REM  ---------------
16001 REM  --- LAUTSTAERKE
16002 REM  ---------------
16010 :
16020 IF A$="▒" THEN L=L+(L>0)
16030 IF A$=CHR$(20) THEN L=L-(L<15)
16040 POKE SI(1)+24,L:PRINT"▒▒▒▒▒▒▒▒▒▒
";TAB(21);
16050 IF L>0 THEN FOR I=1 TO L:PRINT"●";
:NEXT I
16060 IF L<15 THEN FOR I=L TO 14:PRINT".
";:NEXT I
16070 :
16800 RETURN
16900 :
20000 REM"      ┌──────────┐
20001 REM"      │  TOENE   │
20002 REM"      └──────────┘
20010 :
20020 POKE WF(G),I1(IA)-1:P=1
20030 :
20050 A=ASC(A$)-57
20060 IF A<0 THEN A=0
20070 :
20100 T=T%(A):IF T=0 THEN RETURN
20110 N1=M1%(T+D-AK):N2=M2%(T+D-AK)
20120 :
20130 POKE FL(G),N1
20140 POKE FH(G),N2
20150 :
20160 POKE WF(G),I1(IA)
20170 :
20180 GV=G
20200 G=G+1
20210 IF G>H THEN G=1
20220 :
20250 IF AK=0 THEN RETURN
20300 :
20310 IF N1*N2=0 THEN RETURN
20320 N1=M1%(T+D)
20330 IF N1=0 THEN N1=M1%(T+D-SGN(AK))
20340 N2=M2%(T+D)
20350 IF N2=0 THEN N2=M2%(T+D-SGN(AK))
20360 :
20370 GV=G
20380 IF H=1 THEN G=2:GV=1
20390 :
20400 POKE WF(G),I1(IA)-1
20410 POKE FL(G),N1
20420 POKE FH(G),N2
20430 POKE WF(G),I1(IA)
20440 :
20500 G=G+1
20510 IF G>H THEN G=1
20800 :
20900 RETURN
30000 :
50000 END
READY.
```

Listing des Programms »Elektronenorgel« (Schluß)

Schiebung

Dieses kleine Strategiespiel auf dem VC 20 ist die Computerversion eines bekannten »Zeitvertreibers«.

Nach dem Eintippen und Starten des Schiebespiels erscheint ein Feld mit 24 Buchstaben von A bis X und ein Feld mit einem »—«. Es kann nun immer ein Buchstabe eingegeben werden, der neben, über oder unter diesem Feld steht. Anschließend ertönt ein kurzer Piepston und der Buchstabe wandert in das »Leerfeld«, wobei an der Stelle, an der zuvor der Buchstabe war, ein neues Leerfeld entsteht (Buchstabe und »—« wechseln also den Platz). Wird ein Buchstabe eingegeben, der nicht Nachbar des mit »—« bezeichneten Feldes ist, so wird die Eingabe ignoriert.

Dieser Schiebevorgang muß so oft und geschickt wiederholt werden, bis die Buchstaben schließlich in alphabetischer Reihenfolge (spaltenweise von oben nach unten gelesen) sind. Das Leerfeld muß am Ende wieder rechts unten sein.

Wenn dieses geschafft wurde, muß man die Funktionstaste f 1 betätigen. Danach wird geprüft, ob die Aufgabe richtig gelöst wurde. Wenn man schon f 1 drückt, bevor die Reihenfolge korrekt ist, so meldet der Computer den Irrtum und das Spiel kann fortgesetzt werden. Während des Spiels werden rechts unter dem Feld die Anzahl der bisher benötigten Schritte angezeigt.

Das Spiel läuft auf allen Versionen des VC 20 (mit oder ohne Speichererweiterung). (Gerhard Gaber/ev)

Listing zum »Schiebespiel«

```
0 :
1 REM ***************
2 REM *             *
3 REM * SCHIEBESPIEL*
4 REM *             *
5 REM ***************
6 :
7 :
8 GOSUB1000
10 DIMA$(7,7)
30 DIMB$(25)
100 :
110 C=65
120 FORX=1TO5
130 FORY=1TO5
140 A$(X,Y)=CHR$(29)+CHR$(C)
150 C=C+1
160 NEXTY
170 NEXTX
180 A$(5,5)="▓-"
270 PRINT"▓▓▓▓▓"
275 PRINT"┌─┬─┬─┬─┬─┐
276 PRINT"│ │ │ │ │ │
277 PRINT"├─┼─┼─┼─┼─┤
278 PRINT"│ │ │ │ │ │
279 PRINT"├─┼─┼─┼─┼─┤
280 PRINT"│ │ │ │ │ │
281 PRINT"├─┼─┼─┼─┼─┤
282 PRINT"│ │ │ │ │ │
283 PRINT"├─┼─┼─┼─┼─┤
284 PRINT"│ │ │ │ │ │
285 PRINT"└─┴─┴─┴─┴─┘
290 POKE36878,10:T=36876
300 :
310 PRINT"▓"
320 PRINT"▓▓▓▓▓"
330 FORX=1TO5
340 FORY=1TO5
350 PRINTA$(X,Y);
360 NEXT
370 PRINT:PRINT
380 NEXT
390 PRINTTAB(15);PU
400 :
410 GETW$
420 IFW$=""THEN400
440 IFW$=CHR$(133)THEN2000
450 FORX=1TO5
460 FORY=1TO5
470 IFRIGHT$(A$(X,Y),1)=W$THEN500
480 NEXTY,X
500 :
520 IFA$(X-1,Y)<>"▓-" ANDA$(X+1,Y)<>"▓-"
 ANDA$(X,Y-1)<>"▓-" ANDA$(X,Y+1)<>"▓-" T
HEN400
530 POKET,200:FORR=1TO40:NEXT:POKET,0
550 :
570 IFA$(X-1,Y)="▓-"THENA$(0,0)=A$(X,Y):
A$(X,Y)=A$(X-1,Y):A$(X-1,Y)=A$(0,0)
580 IFA$(X+1,Y)="▓-"THENA$(0,0)=A$(X,Y):
A$(X,Y)=A$(X+1,Y):A$(X+1,Y)=A$(0,0)
590 IFA$(X,Y-1)="▓-"THENA$(0,0)=A$(X,Y):
A$(X,Y)=A$(X,Y-1):A$(X,Y-1)=A$(0,0)
600 IFA$(X,Y+1)="▓-"THENA$(0,0)=A$(X,Y):
A$(X,Y)=A$(X,Y+1):A$(X,Y+1)=A$(0,0)
610 PU=PU+1
62[illegible]
630 GOTO 300
1000 :
1020 PRINT"▓;
1040 PRINT"  * SCHIEBESPIEL *
1050 PRINT"▓SIE MUESSEN DURCH       ▓VERSC
HIEBEN DER          ▓BUCHSTABEN IHRE"
1070 PRINT"▓ANORDNUNG SO VER-       ▓AENDE
RN, DASS SIE        ▓STATT WAAGERECHT
1090 PRINT"▓SENKRECHT ZU STEHEN     ▓KOMME
N.
1110 PRINT"▓WENN DAS GESCHAFFT      ▓WURDE
, DANN ▓F1▓         ▓DRUECKEN !   (TASTE)
";
1150 GETW$:
1160 IFW$=""THEN1150
1170 RETURN
2000 :
2030 C=1
2040 FOR X=1TO5
2050 FOR Y=1TO5
2070 B$(C)=A$(Y,X)
2080 C=C+1
2090 NEXT:NEXT
2100 :
2130 FOR C=2TO24
2140 IFB$(C)<B$(C-1)THEN2400
2150 NEXT
2160 PRINT"▓▓▓BRAVO"
2200 END
2220 :
2400 PRINT"▓▓▓DAS WAR LEIDER         ▓ N
ICHT RICHTIG
2430 FORQ=1TO2000
2440 NEXT
2450 PRINT"▓
2480 GOTO270
READY.
```

Bildschirmmasken schnell erstellt

Bei jedem selbstgeschriebenen Anwendungsprogramm steht man in der Regel stets aufs Neue vor dem Problem, zur Abfrage diverser Parameter eine geeignete Bildschirmmake zu erstellen. Dieser Maskengenerator macht die Arbeit etwas einfacher.

Dieser Generator für den VC 20 liest eine Maske direkt vom Bildschirm und erzeugt automatisch die entsprechenden PRINT-Befehle im Programm. Durch diesen Vorgang löscht der Generator sich selbst, so daß ein SAVEn des Programms unmittelbar nach dem Eintippen unbedingt notwendig ist.

Das Programm benötigt eine Erweiterung von mindestens 8 KByte, da am Schluß der Basicspeicher höher gelegt wird. Ohne Erweiterung würde Speicherplatz fehlen. Auch müßte man eine Verschiebung des Bildschirm- und Basicspeicher beachten. Das Programm wird nach dem Laden einfach mit »RUN« gestartet. Danach erscheint eine kurze Anleitung.

So wird die Maske aufgebaut

In Zeile 23 wird der Tastaturpuffer abgefragt. Wurde eine Taste gedrückt, wird er auf 0 zurückgesetzt. Nun wird in Zeile 1000 der Bildschirm gelöscht und eine Datei für den Bildschirm eröffnet, da der Bildschirm dann ja ausgelesen wird und daraus die neuen Zeilen der Maske generiert werden. Sie sehen jetzt eine geänderte Farbe und den blinkenden Cursor. Nun erstellen Sie Ihre Maske nach Ihren Wünschen, wobei Sie mit den Cursortasten bliebig hin- und herfahren können. Ist die Bildschirm-Maske in der richtigen Form, drücken Sie RETURN.

Jetzt wird der Bildschirmspeicher ausgelesen. Die neue Zeile wird mit Zeile 2010 generiert. Das Fragezeichen ist die Kurzform von Print, (CHR$(34) ist der Code für Anführungsstriche. Das Generieren von neuen Programmzeilen geschieht in einer Schleife. Sind alle 23 Bildschirmreihen ausgelesen, springt das Programm nach Zeile 10000. Jetzt wird der Anfang vom Basicspeicher höher gelegt und Zeile 23 gelistet. Nun muß noch ein Leerzeichen aus Zeile 23 entfernt werden. Damit ist die neue Maske fertig und kann abgespeichert werden, oder das nachfolgende Programm kann direkt geschrieben werden. Zeile 50 sorgt dafür, daß das Bild nicht nach oben gescrollt wird. Dadurch wird auch die READY-Meldung unterdrückt. Die fertige Maske wird auch wieder mit RUN gestartet.

(Bernd Borghold/ev)

```
0 PRINT"[…]";
1 PRINT"                              ";
2 PRINT"  |C| \VC-20/ |BILD -|  ";
3 PRINT"  |B| |\___/| |      |  ";
4 PRINT"  |M| | | | | |SCHIRM|  ";
5 PRINT"  | | | |V| | |      |  ";
6 PRINT"  | | | |C| | |MASKEN|  ";
7 PRINT"  |V| | | | | |      |  ";
8 PRINT"  |C| \ |2| / | GENE-|  ";
9 PRINT"  | |  \|0|/  |      |  ";
10 PRINT" |2|   ___    |RATOR |";
11 PRINT"  |0 \_______/         |";
12 PRINT"  |--------------------|";
13 PRINT"  |--------------------|";
14 PRINT"                         ";
15 PRINT"|DIESES PROGRAMM ER-  |";
16 PRINT"|STELLT EIN EIGENES   |";
17 PRINT"|PROGRAMM. NACH DEM   |";
18 PRINT"|ERSTELLEN DER MASKE  |";
19 PRINT"|ENTFERNEN SIE BITTE  |";
20 PRINT"|EIN LEERZEICHEN AUS  |";
21 PRINT"|ZEILE 23 ____________|";
22 PRINT"|________              ";
23 PRINT" *DRUECKE EINE TASTE*";:POKE198
,0:WAIT198,1:POKE198,0:GOTO500
50 GOTO50
500 POKE36879,58
1000 PRINT"[…]";:OPEN1,0:INPUT#1,Q$:CLOSE1
1010 FORT=0TO505:POKE7680+T,PEEK(4096+T)
1020 POKE4096+T,PEEK(7680+T)+128
1030 NEXT:PRINT"[…]"
1035 Y=1
1040 IFY=24THEN10000
1050 FORT=0TO21
1060 C=PEEK(7680+T+(Y-1)*22)
1070 GOSUB6050
1080 ZN$=ZN$+CHR$(C):NEXT
2000 PRINT"[…]"
2010 PRINTY" ?"CHR$(34)ZN$CHR$(34)";"
2020 PRINT"1035 Y="Y+1
2030 PRINT"RUN 1035"
2034 PRINT"[…]ES WIRD GENERIERT:"
2035 PRINT"[…]BILD-ZEILE :#";Y
2040 PRINT"[…]";
2050 POKE631,13:POKE632,13:POKE633,13:PO
KE198,3
3000 END
6050 IFC<32THENC=C+64:RETURN
6051 IFC=34THENC=32
6060 IFC>95THENC=C+64:RETURN
6070 IFC>63THENC=C+32:RETURN
6075 IFC>255THENC=32
6080 IFC>128THENC=C-128:RETURN
6090 RETURN
7000 FORT=0TO505
7010 POKE 4096+T,PEEK(7680+T)+128:NEXT
7100 GOTO7100
10000 POKE5341,0:POKE5342,0:POKE5343,0:P
OKE45,222:POKE46,20
10040 POKE36879,27
10050 PRINT"[…]     MASKE FERTIG !!"
10100 PRINT"[…]ENTFERNEN SIE NUN EIN
             LEERZEICHEN AUS ZEILE"
10120 PRINT"[…]NUMMER 23
10180 FORT=0TO2500:NEXT
10190 PRINT"[…]":LIST23
10200 END
READY.
```

Der Maskengenerator für den VC 20



Deutscher Zeichensatz für den VC 20

Wenn man die ersten (Programmier-) Schritte mit seinem neu erworbenen VC 20 gegangen ist und schon das eine oder andere Programm »geboren« hat, vielleicht ein Adressenprogramm oder eine Lagerliste, spätestens dann wird man die Entdeckung machen, daß der VC 20 trotz deutscher Produktionsstätte nur Englisch spricht, für deutsche Umlaute und »ß« also kein Interesse zu haben scheint.

Dies bedauert man noch umso mehr, wenn man eine anschließbare elektronische Schreibmaschine mit deutschem Typensatz besitzt.

Zwar ist auf der Tastatur des VC 20 kein Platz für einen deutschen Zeichensatz vorgesehen, es ist jedoch möglich, bestimmte Tasten dafür auszuwählen, deren CHR$-Code demjenigen des gewünschten deutschen Umlautes auf der Schreibmaschine entspricht.

Auf diese Weise kann man zwar deutsche Texte drucken, aber auf dem Bildschirm sieht ein solcher Text recht merkwürdig aus: Hier steht zum Beispiel das Zeichen für das britische Pfund statt »ö«.

Auch hier ist mit einigem Know-how Abhilfe zu schaffen oder mit meinem Programm »DEUZEI«.

Der Zeichengenerator

Woher weiß der VC 20 eigentlich, wie die einzelnen Zeichen auszusehen haben, die man auf dem Bildschirm sieht? Hierzu bedient er sich eines Speicherbereichs Hex. 8000 bis 8FFF beziehungsweise 32768 und 36863 dezimal.

In diesem ROM-Bereich sind in genau festgelegter Reihenfolge (vergleiche Bildschirm-Code-Tabelle des Handbuches) die Bitmuster aller verfügbaren Zeichen in jeweils acht hintereinander stehenden Speicherstellen festgelegt.

Wie das funktioniert, veranschaulicht Bild 1: Jedes Zeichen ist auf dem Bildschirm in einer 8 x 8-Matrix wiedergegeben, ein Bildpunkt entspricht einer »1« im Dualcode, sonst steht die »0« (freie Felder). Da ja pro Speicherplatz 8 Bit gespeichert werden, benötigt man 8 Byte pro Zeichen.

Der Computer weiß durch sein Betriebssystem, welche acht Speicherstellen er bei einem Tastendruck abfragen muß, es erscheint soweit recht leicht möglich, ihm ein geändertes Bitmuster in diese Speicherstellen zu »schmuggeln«, zum Beispiel durch POKEs. Dies scheitert allerdings daran, daß der Zeichengenerator im ROM-Bereich liegt und somit nur Lesen, aber nicht Verändern der gespeicherten Daten zuläßt.

Hier hat dankenswerterweise Commodore ein Hintertürchen offengelassen, um doch noch am Datensatz manipulieren zu können. Man kann dem Betriebssystem mitteilen, daß der Zeichensatz an einer anderen Stelle sitzt als normal; dies ist in Speicherstelle 36869 codiert.

```
10 REM ******************************
20 REM *                            *
30 REM *   DEUTSCHER  ZEICHENSATZ   *
40 REM *   FUER VC-20 BAND/FLOPPY   *
50 REM *                            *
60 REM *P. WILKNITZ KARL-CHRIST-STR. 18
70 REM *     6900 HEIDELBERG  -1984- *
80 REM *                            *
90 REM ******************************
100 REM * ZEICHENSATZ  - TRANSFER *
110 FORMP=1TO35
120 READDA%:POKE699+MP,DA%:NEXT
130 POKE36869,236:POKE648,24:CLR
140 SYS700
150 FORT=1TO35:READDA%:NEXT
190 :
200 REM * NEUE ZEICHEN *
210 FORAG=0TO7:READDA%:POKE4824+AG,DA%:NEXT
220 FOROG=0TO7:READDA%:POKE4832+OG,DA%:NEXT
230 FORUG=0TO7:READDA%:POKE4840+UG,DA%:NEXT
240 FORAK=0TO7:READDA%:POKE4312+AK,DA%:NEXT
250 FOROK=0TO7:READDA%:POKE4320+OK,DA%:NEXT
260 FORUK=0TO7:READDA%:POKE4328+UK,DA%:NEXT
270 FORSS=0TO7:READDA%:POKE4336+SS,DA%:NEXT
300 FORAGI=0TO7:READDA%:POKE5848+AGI,DA%:NEXT
310 FOROGI=0TO7:READDA%:POKE5856+OGI,DA%:NEXT
320 FORUGI=0TO7:READDA%:POKE5864+UGI,DA%:NEXT
330 FORAKI=0TO7:READDA%:POKE5336+AKI,DA%:NEXT
340 FOROKI=0TO7:READDA%:POKE5344+OKI,DA%:NEXT
350 FORUKI=0TO7:READDA%:POKE5352+UKI,DA%:NEXT
360 FORSSI=0TO7:READDA%:POKE5360+SSI,DA%:NEXT
390 :
400 REM * FINALE *
410 POKE657,128
420 X=1:PRINT"[illegible]ONDERZEICHEN:":PRINT:P
RINT
430 PRINT"▒ [  \  ]  +  ▒  |  ^":PRINT
440 PRINT"▒ ▒[  \  ]  +  ▒  |  ^"
450 PRINT"[illegible]S DARF GELADEN WERDEN"
490 :
500 REM * DATENBANK *
510 DATA169,0,133,251,169,136,133,252,169,0,13
3,253,169,16,133,254
520 DATA162,8,160,0
530 DATA177,251,145,253,136,208,249,230,252,23
0,254,202,208,242,96
540 DATA90,36,66,126,66,66,66,0
550 DATA90,36,66,66,66,36,24,0
560 DATA90,66,66,66,66,66,60,0
570 DATA36,0,56,4,60,68,58,0
580 DATA36,0,60,66,66,66,60,0
590 DATA36,0,66,66,66,70,58,0
600 DATA60,66,66,124,66,66,124,64
700 DATA165,219,189,129,189,189,189,255
710 DATA165,219,189,189,189,219,231,255
720 DATA165,189,189,189,189,189,195,255
730 DATA219,255,199,251,195,187,197,255
740 DATA219,255,195,189,189,189,195,255
750 DATA219,255,189,189,189,185,197,255
770 DATA195,189,189,131,189,189,131,191
READY.
```

Listing 1. Basiclader für »DEUZEI«. Die Zeichen »\« und »^« in den Zeilen 430 und 440 bedeuten »£« und »↑«.

Benutzt man nun einen RAM-Bereich für den alternativen Zeichensatz, so kann man nach Lust und Laune Veränderungen der Bitmuster vornehmen.

Da man ja doch wohl gern die Bitmuster der meisten Zeichen übernehmen möchte, wäre es ganz sinnvoll, vor der Kreation einiger neuer Zeichen erst einmal die Bitmuster des Zeichengenerator-Bereichs in den ausgewählten RAM-Bereich zu kopieren (immerhin 4 KByte). Diese Aufgabe übernimmt ein kleines Maschinenprogramm (Listing 2), das im Programm »DEUZEI« integriert ist. Ein Kopieren durch Basic-POKEs ist hingegen recht zeitraubend.

Um Speicherplatz im RAM zu sparen, kopiert »DEUZEI« al-

lerdings nur den Zeichenset 2, der Groß- und Kleinbuchstaben, Ziffern und einige Grafikzeichen enthält (normal und invers), aber nur 2 KByte RAM beansprucht.

»DEUZEI« benutzt den RAM-Speicher wie folgt: Zeichengenerator: Page 16 bis 23; Bildschirmspeicher: Page 24 und 25; Basicprogramme ab Page 26 (Bild 2). Eine Page entspricht einem 256-Byte-Block. Der Bildschirmspeicher befindet sich beim VC 20 ab 8 KByte Erweiterung normalerweise in Page 16 und 17 (4096 bis 4607), muß bei »DEUZEI« allerdings »umziehen«; dies erfährt das Betriebssystem durch entsprechende Codierung der Speicherstellen 648 (siehe Zeile 130) und 36869 (zugleich Pointer für den Zeichengenerator).

Das eigentliche Programm »DEUZEI« beginnt bei der Speicherstelle 6656 (Page 26). Da man es nach dem Generieren des neuen Zeichensatzes ja nicht mehr benötigt, wird es beim Einladen eines neuen Programms ab Page 26 gelöscht. Es fehlen dem neuen Programm also nur genau 2 KByte RAM-Speicherplatz, den der neue Zeichensatz beansprucht.

»DEUZEI« intern

Zeile 110-120:	Ab Speicherstelle 700 wird das Maschinenprogramm (35 Byte) abgelegt (unbenutzter RAM-Bereich).
Zeile 130:	Pointer für neuen Zeichengenerator-Bereich und Bildschirm-Bereich.
Zeile 140:	Zeichen-Set 2 wird durch Starten des Maschinenprogramms ab Page 16 abgelegt.
Zeile 210-270:	Bitmuster für deutsche Umlaute wird in neuen Zeichensatz gePOKEd.
Zeile 300-360:	Bitmuster für Umlaute (invers)
Zeile 410:	Umschaltsperre zum anderen Zeichen-Set (nur ein Zeichen-Set existiert im neuen Zeichengenerator-Bereich).
Zeile 420-450:	Bildschirm-Display (neue Zeichen erscheinen)
Zeile 510-530:	Data für Maschinenprogramm
Zeile 540-600:	Data für Umlaute Ä,Ö,Ü,ä,ö,ü,ß
Zeile 700-760:	Data für inverse Umlaute Ä,Ö,Ü,ä,ö,ü,ß

CHR$-Codes:
17 = Cursor nach unten
18 = Inverse Zeichen an
28 = rot
31 = blau
147 = clr home
156 = purpur

Eingabe von »DEUZEI«

Wenn man nun das Programm »DEUZEI« in den Computer eingibt und übereifrig mit RUN startet, wird man eine böse Überraschung erleben: Das Programm stürzt sofort ab und ist zudem auch noch gelöscht!

Des Rätsels Lösung: Normalerweise werden Basicprogramme ja ab Page 18 (Speicherstelle 4608 = 18*256) abgelegt. In diesen Bereich transformiert allerdings das Maschinenprogramm den neuen Zeichensatz und »zerstört« frühzeitig das Basicprogramm. Man muß folglich Sorge tragen, daß das Programm ab Page 26 im Speicher steht. Hierzu sollte man folgendermaßen vorgehen:

1. »DEUZEI« eingeben und auf Band oder Floppy abspeichern (kein RUN!)
2. Programm mit NEW löschen und folgende Basiczeile eingeben:
1 POKE 44,26:POKE 6656,0:PRINT CHR$(3):RUN
3. RETURN-Taste drücken, aber nicht RUN!
4. POKE 44,26:POKE 6656,0 eintippen, dann RETURN-Taste drücken
5. Eingabe von NEW; RETURN-Taste
6. Einladen des abgespeicherten »DEUZEI«

```
02BC  A9 00       LDA #$00      Low-byte Zeichenbereich laden
02BE  85 FB       STA $FB       speichern in dez. 251
02C0  A9 88       LDA #$88      High-byte Zeichenbereich laden
02C2  85 FC       STA $FC       speichern in dez. 252
02C4  A9 00       LDA #$00      Low-byte neuer Z.-Bereich laden
02C6  85 FD       STA $FD       speichern in dez. 253
02C8  A9 10       LDA #$10      High-byte neuer Z.-Bereich lad.
02CA  85 FE       STA $FE       speichern in dez. 254
02CC  A2 08       LDX #$08      X-Register: Zahl der Blöcke
02CE  A0 00       LDY #$00      Y-Register Null setzen (Zähler)
02D0  B1 FB NEXT  LDA($FB),Y    Laden von Koppeladr. (i.indiz.)
02D2  91 FD       STA($FD),Y    Speichern in Koppeladresse
02D4  88          DEY           Zähler -1
02D5  D0 F9       BNE NEXT      Y nicht 0, dann Sprung zu NEXT
02D7  E6 FC       INC $FC       High-byte Z.-Bereich +1
02D9  E6 FE       INC $FE       High-byte neuer Z.-Bereich +1
02DB  CA          DEX           Block-Zähler -1
02DC  D0 F2       BNE NEXT      X nicht 0, dann Sprung zu NEXT
02DE  60          RTS           zurück zu Basic
```

Listing 2. Assembler-Listing zu »DEUZEI«. Das Assemblerprogramm ist im Basiclader nach Listing 1 bereits enthalten und braucht nicht zusätzlich eingegeben zu werden.

7. POKE 44,18 eintippen und unter neuem Namen abspeichern.

Die kleine Mühe hat sich sicherlich gelohnt, denn das neu abgespeicherte Programm ist nun absolut »wartungsfrei«. Es kann ganz normal eingeladen werden, setzt automatisch die Basicuntergrenze auf Page 26, generiert den neuen Zeichensatz in 2 bis 3 Sekunden und erlaubt das Einladen beliebiger Basicprogramme ohne Zusatz-POKEs.

Erklärung für obige Prozedur: Die erste Basiczeile muß am Anfang von Page 18 liegen, der Interpreter erwartet das so. Startet man das »umgebaute« Programm, so wird in dieser Zeile der neue Basicbeginn mitgeteilt, und zwar mit POKE 44,26.

Dieses Miniprogramm wird mit PRINT CHR$(3) abgebrochen und mit RUN in der gleichen Zeile das eigentliche »DEUZEI« ab Page 26 gestartet. Dies sitzt bereits richtig, da man es (siehe unter 4) gleich in Page 26 geladen hat. Danach wurde der Basicanfang wieder durch POKE 44,18 herabgesetzt und somit das funktionsfähige Programm abgespeichert.

Dieses Vorgehen bedeutet zwar, daß beim Abspeichern fast 2 KByte unbrauchbares Material mit abgespeichert wird, aber der Vorteil der besonderen Bedienungsfreundlichkeit überwiegt doch eindeutig.

Arbeiten mit »DEUZEI«

Nach dem Einladen des — wie oben beschrieben bearbeiteten — Programms »DEUZEI« erscheinen die neuen Sonderzeichen auf dem Bildschirm normal/invers, und beliebige Programme können geladen werden.

Die neuen Zeichen und ihr CHR$- beziehungsweise POKE-Code:

	ä	ö	ü	Ä	Ö	Ü	ß
CHR$:	91	92	93	123	124	125	94
POKE:	27	28	29	91	92	93	30
statt:	[	£	]	SHIFT [	SHIFT £	SHIFT]	↑

Dieser Zeichensatz bleibt erhalten, solange der VC 20 eingeschaltet ist, wird allerdings durch Betätigen der RUN STOP/RESTORE-Tasten abgeschaltet; es wird wieder der Pointer auf den normalen Zeichensatz gesetzt. Durch Eingabe von: POKE 36869,236:POKE 648,24:CLR kann man aber wieder den neuen Zeichensatz verfügbar machen. Leider sind sich offensichtlich nicht alle Produzenten von Drucker-Interfaces bezüglich der CHR$-Codes für die deutschen Um-

laute einig. So kann es vorkommen, daß bei bestimmten Druckern das »ß« nicht gedruckt wird, wenn es auch noch so deutlich am Bildschirm prangt. Das ist allerdings kein Beinbruch, man muß nur herausfinden, welchen CHR$-Code dieses Zeichen vom Hersteller erhalten hat. Eine Anpassung von »DEUZEI« ist dann mit Hilfe der Tabellen des VC 20-Handbuchs einfach:

1. In CHR$-Tabelle Zeichen suchen, das der Code-Nummer entspricht.
2. In POKE-Tabelle den POKE-Wert für dieses Zeichen ablesen.
3. POKE-Wert*8+4096 ergibt erste Speicherstelle für neues Zeichen; diesen Wert im Listing ersetzen.

Beispiel: Buchstabe »ß« ist als CHR$(94) vorgesehen.

Normales Zeichen (Pfeil nach oben) entspricht POKE 30. 30*8+4096 = 4336. In Zeile 270 des Listings ist dieser Wert zu finden. Ist nun der CHR$-Code beispielsweise für »ß« = 126 (entspricht dem Zeichen π), so ergibt sich aus der POKE-Tabelle: 94*8+4096 = 4848 als neuer Wert im Listing.

Um die entsprechenden Speicherplatzdaten für die inversen Zeichen zu finden, muß man nur zu den berechneten Werten noch 1024 addieren (vergleiche DATA-Zeilen 700 bis 760 in Listing 1).

»DEUZEI«-Ausbau

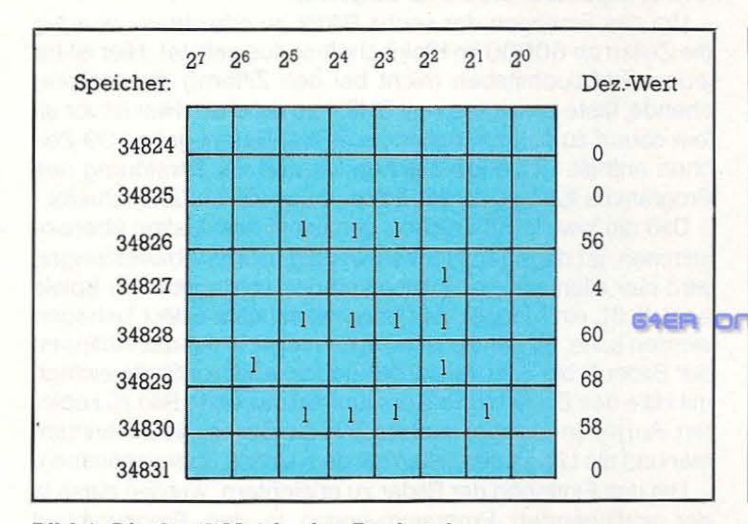
Wem die deutschen Sonderzeichen nicht ausreichen für seine verschiedenen Vorhaben, der kann natürlich recht einfach in das Programm noch zusätzliche »Phantasiezeichen« einbauen. Das Auffinden der gewünschten Speicherstelle und die Berechnung der 8 Byte-Werte für das Bitmuster der 8x8-Zeichenmatrix sollten ja keine Schwierigkeiten mehr bereiten.

Als Beispiel sei der Ersatz des »Klammeraffen« (CHR$:64; POKE:0) durch eine »Brezel« aufgezeigt, die ab Speicherstelle 0*8+4096 = 4096 codiert wird (Bild 3). Diese beiden Programmzeilen wären hier einzufügen:

```
361 forbr=0to7:readda%:poke4096+br,da%:next
770 data0,60,66,165,153,153,102,0
```

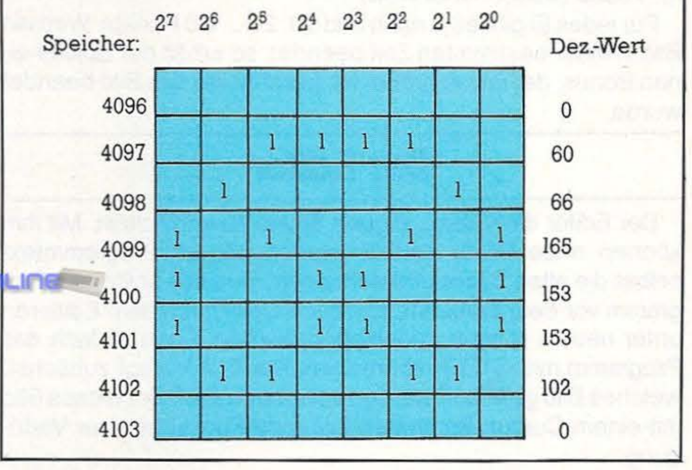
In analoger Weise kann man sich auch leicht beliebige andere Zeichen definieren.

(Peter Wülknitz/ev)



Speicher:	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	Dez.-Wert
34824									0
34825									0
34826			1	1	1				56
34827						1			4
34828			1	1	1	1			60
34829		1				1			68
34830			1	1	1		1		58
34831									0

Bild 1. Die 8 x 8-Matrix des Buchstabens »a«



Speicher:	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	Dez.-Wert
4096									0
4097			1	1	1	1			60
4098		1					1		66
4099	1		1			1		1	165
4100	1			1	1			1	153
4101	1			1	1			1	153
4102		1	1			1	1		102
4103									0

Bild 3. Die 8 x 8-Matrix des Grafiksymbols »Brezel«

	ROM	
Zeichen-Set 2	136-143	34816-36863
Set 1	128-135	32768-34815
Basic-RAM	18-127	4608-32767
TV-Bereich	16+17	4096-4607
3 KByte Erweit.	4-15	1024-4095
1 KByte RAM (benutzt)	0-3 Page	0-1023

	ROM	
Zeichen-Set 2	136-143	34816-36863
Set 1	128-135	32768-34815
Basic-RAM	26-127	6656-32767
TV-Bereich	24+25	6144-6655
Zeichen-Set (neu)	16-23	4096-6143
3 KByte Erweit.	4-15	1024-4095
1 KByte RAM (benutzt)	0-3 Page	0-1023

Bild 2. VC 20-Speicher ab 8 KByte Erweiterung

Spring Vogel, spring

Das Listing des Monats zur Erzeugung von »Jump and Run«-Spielen für Commodore 64.

Bei dem ersten Start des Progrmms dauert es einige Zeit (bei weiteren Starts geht es dann schneller), bis sich das Programm mit der Frage »Edit, Wahl oder Spiel?« meldet.

Mit »W« kann man eines der sechs Bilder zum Spielen auswählen, mit »S« wird ein vollständiges Spiel ab Bild 1 gewählt.

Die Funktionstasten haben folgende Bedeutung:
f1: Dieses Spiel aufgeben, zurück zum Menü
f2: diesen Vogel opfern und mit dem nächsten Vogel neu an der Startposition beginnen
f3: mit dem nächsten Bild weitermachen
f5: Pause (weiter mit SPACE)

Für jedes Ei gibt es je nach Bild 10, 20 ... 60 Punkte. Wird ein Bild in einer bestimmten Zeit beendet, so erhält der Spieler einen Bonus, der um so größer ist, je schneller das Bild beendet wurde.

Der Editor

Der Editor dient dazu, eigene Spiele zu entwickeln. Mit ihm können neue Bilder erstellt werden, die im Programmtext selbst die alten Bilder überschreiben. Deshalb sollte das Programm vor dem Editieren, das neue Spiel nach dem Editieren unter neuem Namen gespeichert werden. Dazu einfach das Programm mit »STOP« abbrechen. Der Editor fragt zunächst, welches Bild geändert werden soll. Dann erscheint dieses Bild mit einem Cursor. Es stehen folgende Funktionen zur Verfügung:
←: die vorgenommenen Änderungen endgültig in den Programmtext übernehmen
f1: Editor verlassen, ohne Änderungen zu übernehmen. Das Bild bleibt erhalten, wie es vor dem Editoraufruf war. In beiden Fällen fragt das Programm, ob das Bild ausprobiert werden soll. Aus dem Spiel gelangt man mit f1 und E jederzeit in den Editor zurück.

Die Cursorsteuerung erfolgt wie gewohnt über die Cursortasten. SHIFT CLR: Bild löschen

Die Startposition des Vogels wird durch »0« festgelegt. In jedes Bild können zwischen 0 und 7 Monster gesetzt werden, die sich horizontal auf einer Breite von acht Spalten hin und her bewegen. Das linkte Ende dieser Bewegungsbahn kann mit den Tasten 1 bis 7 festgelegt werden.

Mit den Tasten A bis T können die 20 verschiedenen Bausteine, die zur Verfügung stehen, gesetzt werden. Der Vogel reagiert immer nur auf das Zeichen unter seinen Füßen. Bei Sprüngen reagiert er (außer bei Wänden) erst in der absteigenden Phase wieder auf Zeichen. Eine Übersicht über die Bausteine bietet die Tabelle 1. Dazu einige ergänzende Bemerkungen:

Die Bausteine A bis D sind durch verschiedene Bewegungsmöglichkeiten gekennzeichnet. Springen ist möglich:
A: rechts, links
B: rechts, links, runter
C: rechts, links, rauf
D: alle vier Richtungen

Bei E bis H ist eine Bewegung nur in jeweils eine Richtung und kein Sprung möglich.

Bei I bis N wird der Vogel automatisch bewegt. Er selbst kann nur springen.

Der Trampolin 0 bewirkt einen zufälligen Sprung nach rechts oder links.

Bei der Gummiwand S wird der Vogel auf das letzte Zeichen, das weder Wand noch Luft war, zurückgeworfen.

Hinweise zum Eintippen

Da das Problem in die Interruptroutine eingreift und seinen eigenen Programmtext verändert, sollte es auf jeden Fall vor dem ersten Start abgespeichert werden.

Um das Abtippen des Listings zu erleichtern, wurden im Listing alle Steuerzeichen innerhalb von Printbefehlen durch leicht lesbare Worte entsprechend Tabelle 2 ersetzt. Wenn zum Beispiel im Listing <CD> steht, dann ist an dieser Stellle CURSOR DOWN einzugeben.

Um Tippfehler bei den zahlreichen DATA-Zeilen besser lokalisieren zu können, wurden diese in verschiedene Blöcke mit jeweils eigener Prüfsumme aufgeteilt.

Um das Eintippen der sechs Bilder zu erleichtern, wurden die Zeilen ab 60000 im Kleinschriftmodus gelistet. Hier ist für jeden Großbuchstaben (nicht bei den Ziffern!) die entsprechende Taste zusammen mit SHIFT zu drücken. Hier ist vor allem darauf zu achten, daß jeder PRINT-Befehl genau 39 Zeichen enthält. Fehlende Zeichen können zur Zerstörung des Programms führen, da der Editor diese Zeilen überschreibt.

Daß die jeweils 39 Zeichen genau mit dem Listing übereinstimmen, ist dagegen nicht so wichtig. Durch Abweichungen wird hier allenfalls die Spielbarkeit des vorgegebenen Spiels beeinflußt, ein Mangel, der jederzeit mit dem Editor behoben werden kann. Im Prinzip ist es auch möglich, auf das Abtippen der Bilder 2 bis 6 zu verzichten und an anderer Stelle einfach mit Hilfe des Bildschirmeditors fünfmal das erste Bild zu kopieren. Auf jeden Fall aber muß die Zahl der Zeilen, die Zeilennummer und die Länge der Zeilen mit dem Listing übereinstimmen.

Um das Eintippen der Bilder zu erleichtern, wurden diese in der vorliegenden Programmversion in den Programmtext selbst gelegt. Deshalb ist die Zahl der Bilder auf sechs (diese Zahl wurde gewählt, um die Tipparbeit in erträglichen Grenzen zu halten) festgelegt. Die Anzahl der Bilder kann aber folgendermaßen erhöht werden:

Der Variablen BM in Zeile 50070 ist die gewünschte Anzahl zuzuweisen. Für jedes weitere Bild sind dem Programm ent-

TASTE	: BAUSTEIN
A	: BODEN
B	: LEITER UNTERES ENDE
C	: LEITER OBERES ENDE
D	: LEITER MITTELSTÜCK
E	: SEIL HOCH
F	: SEIL RUNTER
G	: EINBAHNSTRASSE RECHTS
H	: EINBAHNSTRASSE LINKS
I	: TRANSPORTBAND RECHTS
J	: TRANSPORTBAND LINKS
K	: AUFZUG HOCH
L	: AUFZUG RUNTER
M	: RUTSCHBAHN RECHTS RUNTER
N	: RUTSCHBAHN LINKS RUNTER
O	: TRAMPOLIN
P	: LUFT, FREIER FALL
Q	: TOEDLICHE FALLE
R	: EI
S	: GUMMI-WAND
T	:MAGISCHE FLÜGEL

Tabelle 1. Die verschiedenen Bausteine und ihre Bedeutung

sprechend dem Schema der ersten sechs Bilder 22 PRINT-Zeilen, die jeweils 39 beliebige Zeichen ausgeben, und ein abschließendes RETURN, anzufügen. Die erste Zeilennummer eines Bildes berechnet sich nach der Formel 59900 + Bildnummer*100 +1, zum Beispiel 60601 für Bild 7.

(Matthias Törk/aa)

Im Listing	: zu drückende Taste
<HO>	:HOME
<CLR>	:CLR
<RON>	:RVS ON
<ROF>	:RVS OFF
<CD>	:CURSOR DOWN
<CU>	:CURSOR UP
<CR>	:CURSOR RIGHT
<CL>	:CURSOR LEFT
<F1>...<F8>	:FUNKTIONSTASTEN
<BLK>...<YEL>	:FARBEN 0-7
[1]...[8]	:FARBEN 8-15, ZIFFER + COMMODORE-TASTE

Tabelle 2. Darstellung der Steuerzeichen im Listing

Die wichtigsten Variablen	
XS,YS	: Spritekoordinaten des Helden
XG,YG	: Koordinatengrenzen
OX,OY	: Koordinatenoffset Sprite/Zeichen
TG	: Zeitlimit
BM	: maximale Bildzahl
BI	: aktuelles Bild
UP ( ), DO ( ),RI ( ), LE ( ),DX ( ), DY ( )	: Bewegungstabellen
JU ( )	: Sprungtabelle
CO ( )	: Farbtabelle
JP,JS	: Joystick
EZ,EM	: Anzahl Eier
SC	: Punkte
MZ	: Anzahl Leben
CP	: Cursorposition
BS	: Bildschirmadresse
FR	: Adresse Farb-RAM
KO	: Sprite-Kollisionsregister
V	: VIC-Chip
ZS	: Adresse Zeichensatz
EA	: Adresse Bilder
SI,W1,GL,GH	: Sound

Tabelle 3. Die wichtigsten Variablen von Spring-Vogel

Die wichtigsten Routinen	
50000:	INITIALISIERUNG
48000:	MENUE
49000:	EDITOR
45000:	AB BILD BI SPIELEN
1050:	EIN BILD SPIELEN
46000:	BILD BI INITIALISIEREN
500:	JOYSTICK
600:	FIGUREN BEWEGEN
40000:	SPRINGEN
11000:	EI
2000:	LUFT
12000:	TOEDLICHE FALLE
15000:	FLIEGEN

Tabelle 4. Die wichtigsten Routinen nach Zeilennummern aufgeschlüsselt

```
10 REM *** SPRING-VOGEL 9.3.LIST
11 REM *** M.TOERK       17.6.84
20 POKE53280,0:POKE53281,0
25 PRINT"<CLR>NICHT WUNDERN UND WARTEN!!
"
30 GOSUB 50000:GOTO 48000
499 REM ** JOYSTICK **
500 JS=PEEK(JP):IF (JSAND16)=0 THEN GOSU
B 40000:GOTO 600
505 IF (JSANDQ1)=Q0THEN POKEW1,SW:IF UP(
CM) THEN YS=YS-Q8:GOTO 590
510 IF(JSANDQ2)=Q0THEN POKEW1,SW:IF DO(C
M) THEN YS=YS+Q8:GOTO590
520 IF(JSANDQ4)=Q0THEN POKEW1,SW:IF LE(C
M) THEN XS=XS-Q4:GOTO590
530 IF(JSANDQ8)=Q0 THEN POKEW1,SW:IF RI(
CM) THEN XS=XS+Q4
590 POKEW1,Q0
600 IFXS<OXTHENXS=OX
620 SYS(SR) Q0,XS,YS:RETURN
1000 REM *** MAIN LOOP **
1050 IF PEEK(TA)THEN1200
1055 IF PEEK(KO)ANDQ1 THEN CM=16:POKEKO,
0:GOTO1100
1060 CM=USR(XS-OX) ,YS-OY:IF CM=Z8 THEN
XS=8*PEEK(49426)+OX:YS=8*PEEK(49427)+OY
1070 IF CM>Q7 THEN XS=XS+DX(CM):YS=YS+DY
(CM)
1075 IF TI<TG THEN PRINT"<HO>",INT(TI/10
0)
1080 IF CM<Z4 THEN GOSUB 500:GOTO 1050
1100 ON CM-13 GOTO 3000,2000,12000,11000
,1050,15000
1200 GETB$:IFB$="<F1>"THEN RETURN
1210 IF B$="<F3>" THEN CM=16:GOTO 1100
1215 IF B$="<F5>" THEN 11040
1217 IF B$="<F7>" THEN POKE 198,0:WAIT19
8,1
1220 GOTO1050
2000 I=0:POKEW1,33
2010 I=I+1:YS=YS+4:IF I>8 OR YS>YG THEN
CM=16:POKEW1,0:POKESI+1,GH:GOTO 1100
2015 POKESI+1,40-2*I
2020 GOSUB 600:CM=USR(XS-OX) ,YS-OY:IF C
M=15 THEN    2010
2025 IFCM=18THENXS=FNX(XS)
2030 POKEW1,0:POKESI+1,GH:YS=FNY(YS):GOT
O 1050
3000 RX=2-4*INT(RND(1)*2):GOSUB40100:GOS
UB600:GOTO1050
10999 REM ** GEGENSTAND POSITIV, EI**
11000 POKE BS+40*PEEK(49427)+PEEK(49426)
,80:SC=SC+BI*10:EZ=EZ+1
11030 IF EZ<EM THEN PRINT"<HO>",,SC:GOTO
1050
11035 IF TI<TG THEN SC=SC+INT((TG-TI)/10
)
11040 BI=BI+1:IF BI>BM THEN RETURN
11050 GOSUB 46000:GOTO 1050
12000 MZ=MZ-1:IF MZ=0 THEN RETURN
12050 PRINT"<HO>",,SC,MID$("******
",7-MZ,6)
12052 POKEW1,17
12055 FOR XS=XS TO XX STEP SGN(XX-XS):GO
SUB 600:POKESI+1,49+(XSAND1):NEXT
12057 FOR YS=YS TO YY STEP SGN(YY-YS):GO
SUB 600:POKESI+1,52+(YSAND1):NEXT
12060 XS=XX:YS=YY:POKEKO,0:POKEW1,0
12070 GOTO 1050
15000 POKEV+29,255:POKE BS+40*PEEK(49427
)+PEEK(49426),80:CM=3:XS=XS-10:SW=17
15050 JS=PEEK(56320)
```

Listing »Spring-Vogel«

```
50040 XH=49400:XL=49392:TY=49432
50042 DEFFNY(Y)=PEEK(49427)*8+OY
50045 DEFFNX(X)=PEEK(49426)*8+OX
50050 V=53248:BS=256*204:XG=320+OX:YG=20
0+OY
50060 KO=V+30:FR=55296:REM SPR.KOLLIS.RE
G / FARBRAM
50070 POKE 786,193:POKE 785,32:BM=6: REM
 USR ADR.,MAX.BILDER
50090 DIM JU(13):FORI=0TO13:READJU(I):NE
XT
50100 DATA 5,4,3,2,1,1,0,0,-1,-1,-2,-3,-
4,-5
50215 FORI=BS+1017 TO BS+1023:POKEI,16:N
EXT:POKEBS+1016,14
50220 POKE V+23,0:POKE V+29,254:POKEV+39
,7
50230 DIMX(7),Y(7),RX(7),RY(1):FORI=0TO7
:X(I)=100:Y(I)=100:NEXT
50249 REM MAPRO-TABELLEN
50250 FORI=1TO7:POKEV+39+I,I:POKE49432+I
,30*I:POKE49392+I,35*I:POKE49400+I,0
50251 POKE49416+I,1:POKE49408+I,5+RND(1)
*45:NEXT:POKE49401,1
50252 POKE 49424,1:REM SB-FLAG
50410 DIM UP(20),DO(20),RI(20),LE(20),DX
(20),DY(20)
50422 RI(0)=1:LE(0)=1:RI(1)=1:LE(1)=1:UP
(1)=1:RI(2)=1:LE(2)=1:DO(2)=1
50437 UP(3)=1:DO(3)=1:RI(3)=1:LE(3)=1:UP
(4)=1:DO(5)=1:RI(6)=1:LE(7)=1:DX(8)=4
50477 DX(9)=-4:DY(10)=-8:DY(11)=8:DY(12)
=8:DX(12)=8:DY(13)=8:DX(13)=-8:DY(15)=4
50599 REM SOUND
50600 SI=54272:W1=SI+4
50620 POKE SI+24,15
50625 POKE SI+5,0:POKE SI+6,15*16+0
50630 GL=180:GH=18:A=0
50710 FORI=13TO16:FORN=0TO62:READ Q:A=A+
Q:POKE 49152+I*64+N,Q:NEXT:NEXT
50720 IF A<>16572 THEN PRINT"DATAERROR":
STOP
50800 DATA0,255,0,1,255,128,1,153,128,1,
153,128,3,255,192,31,195,248
50805 DATA 63,227,252,103,243,230,195,24
9,195,131,253,193,3,255,192
50807 DATA 3,255,192,1,231,128,0,231,0,0
,102,0,0,102,0,0,102,0,1,231,128
50809 DATA ,,,,,,,,
50810 DATA0,255,0,1,255,128,129,153,129,
193,153,131,99,255,198,63,195,252
50815 DATA 31,199,248,7,207,224,3,159,19
2,3,191,192,3,255,192
50817 DATA 3,255,192,1,231,128,0,231,0,0
,102,0,0,102,0,0,102,0,1,231,128
50819 DATA ,,,,,,,,
50850 DATA 0,192,0, 1,224,0, 3,240,0, 2,
208,0, 2,208,0,3,240,0,3,240,0
50855 DATA 3,48,0,15,112,0
50857 DATA ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,
50870 DATA 0,192,0,1,224,0,3,240,0, 2,20
8,0, 2,208,0, 3,240,0, 3,240,0
50875 DATA 3,48,0,3,188,0
50880 DATA ,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
,,,,,,
51000 DIM CO(19):FOR I=0 TO 19:READ CO(I
):NEXT
51010 DATA 5,5,5,2,10,8,13,13,5,5,4,4,4,
4,12,1,2,7,8,14
52000 REM EDIT-ANFANGSADR.
52010 DEFFNAD(X)=PEEK(X)+256*PEEK(X+1)
52020 EA=2049
52050 IF FNAD(EA+2)<59800 THEN EA=FNAD(E
A):GOTO 52050
58000 REM ZEICHENSATZ NACH $E0000 (ZS),
VIDERRAM NACH $CC00 (BS)
58020 ZS=14*4096:ZV=53248
58040 IF(PEEK(53272)AND254)=56 THEN RETU
RN
58050 POKE 53272,56:POKE 56576,148:POKE
648,204:BS=204*256
58055 PRINT"<CLR>       GEDULD":PRINT:GO
SUB 60400
58060 POKE 56334,0:POKE 1,51
58070 FOR I=0 TO 2047:POKE ZS+I,PEEK(ZV+
I):NEXT
58075 POKE 1,55:POKE 56334,1:A=0
58090 READ C:A=A+C:IF C>255 THEN 59010
58095 H=ZS+8*(C+65):FOR I=0 TO 7:READ C:
A=A+C:POKE H+I,C:NEXT
58098 GOTO 58090
58100 DATA 0,255,60,24,24,24,60,255,255
58110 DATA 1,66,66,66,66,255,24,24,255
58120 DATA 2,255,24,24,255,66,66,66,66
58130 DATA 3,66,66,126,66,66,66,126,66
58140 DATA 4,12,24,48,0,12,24,48,0
58150 DATA 5,48,24,12,0,48,24,12,0
58160 DATA 6,255,129,8,4,62,4,8,255
58170 DATA 7,255,129,16,32,124,32,16,255
58180 DATA 8,255,56,28,14,14,28,56,255
58190 DATA 9,255,28,56,112,112,56,28,255
58200 DATA 10,66,66,90,102,102,66,66,66
58210 DATA 11,66,66,102,102,90,66,66,66
58220 DATA 12,192,208,112,56,28,14,7,3
58230 DATA 13,3,7,14,28,56,112,208,192
58240 DATA 14,255,62,12,48,192,48,12,62
58250 DATA 15,0,0,0,0,0,0,0,0
58260 DATA 16,0,126,126,102,102,126,126,
0
58270 DATA 17,24,60,60,126,126,126,60,24
58280 DATA 18,0,223,223,0,0,251,251,0
58290 DATA 19,0,102,255,153,153,56,56,0
58299 DATA 999
59000 REM ** MAPROS **
59010 IF A<>14182 THENPRINT"DATA-ERROR I
N 58100FF":STOP
59014 A=0:FOR I=49152 TO 49378:READ N:PO
KE I,N:A=A+N:NEXTI
59016 IF A<>27365THENPRINT"DATA ERROR":S
TOP
59020 DATA 80,70,83,32,158,183,224,8,176
,31,189,219,192,45,21,208,141
59022 DATA 21,208,96,32,158,183,224,16,1
76,14,134,2,32,253,174,32,235
59024 DATA 183,165,21,201,2,144,3,76,72,
178,138,72,166,2,32,10,192
59026 DATA 189,219,192,45,16,208,141,16,
208,70,21,144,9,189,211,192,13
59028 DATA 16,208,141,16,208,138,10,170,
104,157,1,208,165,20,157,0,208
59030 DATA 166,2,189,211,192,13,21,208,1
41,21,208,96
59099 REM
59100 DATA 32, 20,192,173, 16,193, 73,25
4,141, 16
59102 DATA193,162, 7,189,248,207, 24,109
, 16,193
59104 DATA157,248,207,202, 16,243,169, 7
,133, 2
59106 DATA166, 2,189, 0,193, 24,125, 8,1
93,157
59108 DATA 0,193,201, 64,144, 11,189, 8,
193, 73
```

Listing »Spring-Vogel« (Fortsetzung)

```
59110 DATA254,157, 8,193,176,232,234, 24
,125,240
59112 DATA192,133, 20, 42, 29,248,192, 4
1, 1,234
59114 DATA234,133, 21,189, 24,193, 32, 4
5,192,198
59116 DATA 2,208,203,173,30,208,96
59140 DATA 253,174,32
59142 DATA 9,185,211,192,13,29,208,141,2
9,208,70,2,144,9,185,211,192
59144 DATA 13,23,208,141,23,208,96,1,2,4
,8,16,32,64,128,254,253
59146 DATA 251,247,239,223,191,127
59190 REM
59195 A=0:FORI=49440TO49505:READN:A=A+N:
POKEI,N:NEXT
59196 IF A<>8224THEN PRINT"DATA ERROR":S
TOP
59200 DATA 32,238,183,102, 21,165, 20,10
6, 74, 74
59202 DATA133,211,201, 40,144, 4,169, 15
,176, 42
59204 DATA133,211,138, 74, 74, 74,133,21
4, 32,108
59206 DATA229,164,211,177,209, 56,233, 6
5,201, 20
59208 DATA144, 2,169, 15,201, 15,240, 14
,201, 18
59210 DATA240, 10,166,211,142, 18,193,16
6,214,142
59212 DATA 19,193,168, 76,162,179
59800 RETURN
59999 REM BILD 1
60001 print"PPPPPPPPPPPPPPPRPPPPPPPPPPPPPPPPPPPPPPPS"
60002 print"PPPPLJJJJAAAAAAAAAQAAAAPPPIIIITPPPPPPRPS"
60003 print"PPPPLPPPPPEPPPPPPPPPPPPPPPPPPPPPPPPPPPPS"
60004 print"PPPPLPPPPPEPPPPPPPPPPPPPPPPPPPPPRPPAAAAS"
60005 print"PPPPLPPPPPEPPPPPPPPPPPPPPPPPPPAAAAAPPPSS"
60006 print"PPRPLPRPPPEPPPPPPPPSSSSSSSSSPPPPPPPPPPSS"
60007 print"PAAACAAAPPEAAAAPPPPPPPPPPPPPPPPPPPPPPRSS"
60008 print"PPPPFPPPPPEPPPPPPPP2PPPPPPPPPPPPPPPPPPSS"
60009 print"PPPPFPPPPPEPPPAAAAAAAAAAAAAAAAAAAOAAAASS"
60010 print"PPPPFPPPPPEPPPRPPPPPPPPPPPPPPPPPPPPPPDPP"
60011 print"PPPPFPPPPPEPPAAAAAPPPPPPPPPPPPPPPPPPPDPP"
60012 print"PPPPFPPPPPEPPPPPPPPPPPPPPPPPPPPPPPPPPDPP"
60013 print"PPP1FPPPPPEPPPPPPPPPPPPPPPPPPPPPPPPPPDPP"
60014 print"PPPPFPPPPPEPPPPPPPPPPPPPPPPPPRHHHHHHABAC"
60015 print"PPPPPPPPPPEPPPPPPPPPRPPPPPP3PPPPPPPPPPPD"
60016 print"PPPPRPPPPPEPPPPPPPPPAPPPPPPPPPPPPPPPPPPD"
60017 print"PPPPGGGGGGEPPPPPPPPRKPPPPPPPPPPPPPPPPPPD"
60018 print"PPPPPPPPPPPPPPPPPPPPPPPPAKAPPPIIPPPPPRPD"
60019 print"PPPPPPPPPPPPPPPPPPPPPPPKPPPPKPRPPPRPPAAB"
60020 print"PPPPPPPPPPPPPPPPPRPPPPPPKPPPPKPPPAAAPPPP"
60021 print"P0PPRPPPPAAAAIIIRPPKPPPPKAAAAPPPPPPPPPPP"
60022 print"AAAAAAAAAASSSSSSSSAAABAIIIKSSSSSSSSSSSSS"
60030 return
60101 print"PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP"
60102 print"PPPPPRPPRPPPPPPPPPPPPPPPPPPPPPRPPPPRPPPP"
60103 print"AARAAAAAAAAPPPAAARPRAAAPPPAAAAAAAAAAARAA"
60104 print"PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP"
60105 print"PPRPPPPPIIIIP4PPPPPPPPPPPJJJPPPPPPPPPRPP"
60106 print"PPPPPPPPPPPPPRJJJAIIIIRPPPPPPPPPPPPPPPPP"
60107 print"PPRIIIIPPPPPPPPPPPPPPPPPPPPPPPPPJJJJPRPP"
60108 print"PPPPPPPJJJPPPPPPPPPPPPPPPPPPIIIPPPPPPPPP"
60109 print"PPRPPPPPPPJJJPPPPPPPPPPPPPIIIIPPPPPPPRPP"
60110 print"PPPPPPPPPPPPPPJJJPPPIIIPPPPPPPPPPPPPPPPP"
60111 print"PP0PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP"
60112 print"CAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA"
60113 print"DSPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP"
60114 print"DSRPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPTP"
60115 print"DSAPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPAP"
60116 print"DSPPP5PPPPRPPPPPRPPPPPPRPP6PRPPPPPPPPPPR"
60117 print"DSAACPPPPCAAAAAAAAAAAAAAAAAAAAAAAACAAAA"
60118 print"DPPPDPPPPDPPPPPPPPPPPPPPPPPPPPPPPPPDPPPP"
60119 print"DTPPDPPPPDPPPPPPPPPPPPPPPPPPPPPPPPPDPPPP"
60120 print"DAPPDPPPPDPPPPRPPRPPPRPPRPPRPPRPPPPDPPPR"
60121 print"DPPPDPPP2DPPPPPPPPPPPPPPP3PPPPPPPPPDPPPP"
60122 print"BAAABAAAABAAAAAAAAAAAAAAAAAAAAAAAAABAAAA"
60150 return
60199 rem bild3
60210 print"SSSSRHHPQPJJJCIIIPPPIIIRIIPPPAAAAAAAAAAR"
60214 print"PPPSFSPPPPSPPDPSSPPQPPPLPPPPPPPPPPPPPPPN"
60216 print"IIRSFSPPPPSPPDPSSPPPPSPLPPPPPPPP0PPPPPNP"
60218 print"KPPSFSPPRPSPPDPSSPPPRSPLPPPPPPPPPAPPPNPP"
60220 print"KPLSFSPPRPSPPDPSSPPAASPLPPPPPPPPPPPPNPPP"
60222 print"KPLSFSPPRPSPPDPSSPPPPPPLPPPPPPPPPPPNPPPP"
60224 print"KPLSFSPPPPSPPDPSSAAPPPPLPPPPPPPPPPNPPPPP"
60226 print"KPLSFSP5PPPPPDPSPP6PPPPLPPPPPPPPPNPPPPPP"
60228 print"KPLSFSPPIILPPDPSPPAAPPPLPPPPPPPPNPPPPPPP"
60230 print"KPLSFSPPPPLPPDPSPPPAAPPLPPPPPPPNPPPPPPPP"
60234 print"KPLSFSPPPPLPPDPSPPPPPPPLPPPPPNPPSPPPPPPP"
60237 print"KPRSFSPPPPLPPDPSAAAPPPPLPPP4NPPPSPPPPPRP"
60240 print"KPPSFSPPPPLPPDPSPPPAAAPLPPPNPPPPSPPPPAAP"
60244 print"KPPSFSPPPPLPPDPSPPPPPKPLPPPRPPPPPSAAAPPP"
60247 print"KPPSFSPPPPIIIDIIIIIIIKPLPPMPPPPPPPPPPPPP"
60250 print"KPTSLSPPPPPPPDPPPPPPPPPLPPPMPPPPPPPPPAAP"
60254 print"KPPSLSPPPPPPPDPPPPPPPPPLPPPPMPPPSPPPP2PP"
60258 print"KJJJJSPPRPP1PDPPPPPPPPPPLPPPPMPPSAAAPPPP"
60260 print"PPPPPPPPAAAPPDPPPPPPPPPLPPPPMPPPPPPPPPPP"
60263 print"PPPPPPPPPPPPPDPPPPPPPPPLPPPPPPPPMPPPPAAA"
60264 print"PPPPPPPPPPPPPDPPPPRPP3PLPPPPPPPPMPPPPPPP"
60270 print"AAAAAAAAAAAAABAAAAAAAAAAAAAAAAAAAAAAAAAA"
60280 return
60299 rem bild4
60301 print"PAAARAAPPPPPPPPPPPPPPPPPPPPPPPSSSPRJJJJJ"
60302 print"PPPPPPPIIIIIIIRIIIIICIIIIIIIILSSNJJ2PPPK"
60303 print"PPPPPPPPPPPPPPPRSSSSSLSEPPPPPLSNPPPPPPPK"
60304 print"PPPIIIIPPPPJJJJSSPPPLSEPPPPPLNPPPPPPPPRK"
60305 print"PPPPPPPPPPPPPP3PPSPPPPLSEPPJAABIPPPIIPPAB"
60306 print"JJJPPPPPPPPPPPPPSARSSLSERPPPPPPPPPPPPPPP"
60307 print"PPPPPPPPPPPPPPPPSSMSSLSEIPPPPPRPPPPPPPPR"
60308 print"PPPPRJJJPPPPPPPPSSSPSLSEPRPPIIIPPPIIPPIC"
60309 print"PPFHHHHHPPJJJPPPPSRSLSEPIPPPPPHHHHHHHD"
60310 print"PPFSPPPPPPPPPPPPPPSPSLSEHHHHHHHPPPPPPPPD"
60311 print"PPFSPPPPPPPPJJJJPSRSLSEPPPPPPPPPPPPPPPPR"
60312 print"PRFSPPPPPPPPPPPPPPSPSLSEPPRPPPPRPPPPPPPJ"
60313 print"PAHSPPPPPPPPPPPIIPSRSLSEPPAPPJJJPPOPPPAP"
60314 print"PPPPPPIIIPIIIPPPPSPSPSPPAHHHHHHHHHHHPPR"
60315 print"PPPPIIIPPPPPPPPPPPPPPPPPPPPSSSSSSSSSSAAA"
60316 print"IIIIIGGGGGGGGGGGGGGGGGGGGGLSSSSSSSSSSSS"
60317 print"SDSSSSSSSSSSSSSSSSSSSSSSSSLSSSPPPPPPPPPS"
60318 print"SDPSSSSSSSSSSSSSSSSSSSSSSSLSSSPPPRPPPPRS"
60319 print"SDRSSSSSSSSSSSSSSSSSSSSSSS0SSSRPPIPPPAS"
60320 print"SDPPPPPPPPPPPPPPPPPPPPPPPPPPPSSSIPPPPPPPS"
60321 print"SDP1PPPPPPPPPPPP4PPPPPPPPPPPPPPPPP5PPPS"
60322 print"ABAAAAAAAAAAAJJJJJJJJJJAAAAAAAAAAAAAS"
60323 return
60399 rem bild5
60401 print"AAARAQJPP1ARAGCHARAPPPPRJJJAAAPCPPPPPPP"
60402 print"PPPPSSSPPPPPPPDPPPPPPPPPPPPPPPPPPKPPPPPPP"
60403 print"PPPPSSSPPPPPPPDPPPPPPPPPPPPPPPPPPKJJPPPPP"
60404 print"PPPNSSSPPPPRPPDPPRPPPPPPMPPPPPPPPPPPPPPP"
60405 print"PPNPSSSIIIIIPPDPPPPPPPPPPPPPPPPPPPAAPPP"
60406 print"PPMPSSSPPPPPQPEPPPPPPPPPPPPPRPPPHHPPPPP"
60407 print"PPPMPSSCAPPPPPEPPJJJCOPPOPPPOPPOPPPPPPP"
60408 print"PPPIMPPDPSPPPPEP3PPPDPPPPPPPPPPPPPAAAPP"
60409 print"RPPPPMPDPSAARABAAPPPDPPPIIIIPJJJJPPPPRP"
60410 print"AAPPPPMDPSPPPPPPPPPPPDPP2PJJJJPPPPPPPPAAP"
60411 print"QJJJPPPDPSAAAAAQJJJJDPPPPPPPPPPPPPIIIPP"
60412 print"PPPPPPPDPSPPPPPPPPPPPDPPPIIIIPPPIOIQPPPP"
60413 print"PQJJPPPDPSPPPPPPPPPPPPDPPPPPPPPPPKSSSSSCP"
60414 print"PPPPPPPDPPPPPPP4PPPPDPPPPPPPPPPKSSSSSLP"
60415 print"PPIIIPPBAAAAAAAAAAAABAAAAAAAAAAAKSSSSSPP"
60416 print"PPPPPPPSSSSSSSSSSSSPQPPKPPLPPKPPLPPPSPS"
60417 print"IIPPPPPSSSSSSSSSPPPPPDPPKPPLPPKPPLAACSPS"
60418 print"PPPPPPPSSSSSSSSSPHHHPDPPKPPLPPKPPLPPDSLS"
60419 print"PPQJJPPPPPPPPPSSRSSSPDPPKPPLPPKPPLPDDSRS"
60420 print"PPPPPPPPPPPPPPSSPSSSPDPPKPPLPPKPPLPDRSPS"
60421 print"IIPPPPPPPPPPPP0PPSSSPDPPKPPLPPKPPLPDPPLS"
60422 print"AAAAAAAAAQJJJJJAAAAIBIQQQQQQQQQQABAAHS"
60480 return
60499 rem bild6
60501 print"PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPQ"
60502 print"PPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPPQJJJPK"
60503 print"PPPPPPPPHHHQJPPPPPPPPPPPPPPPPPPPPPQPPPPK"
60504 print"PPPPPPPRPPPPPPPJJJPPPPPPPPPPPPPPPQPPPPK"
60505 print"PPPP4PPPPPPPPPPPPPPPPPPPPPPPPPPJJJJQQJJJQ"
60506 print"PPPPPPPCPPPPPPPPPPOJJPPPPPPPPRPPPPPPPPPPP"
60507 print"PPPPPPNSMPPPPPPPPPPPPPPPPPJJJQQJPPPPPPIIIQ"
60508 print"PPPPPNSSSMPPPPPPPPPPPPPPPPPPPPPPPPPPPPPP"
60509 print"PPPPNSSPSSMPPPPPPPPPPPPPPPPPPPPPPPPPJJPPP"
60510 print"PPPNSSPRPSSMQQQPQQQPQQQQQMPPPPPPPPPPPPPP"
60511 print"PPNSSSPTPSSSMQQRQQQRQQQQQQPPPPPPPPPIIIIQ"
60512 print"PQSSSSSSSSSSSSSIIIIIIIIIIIIIMPPPPPPPPPSPPK"
60513 print"RPSPRPSSSPPPSSSSSSSSSSSSSSSSMPPPPPPPPQPPK"
60514 print"IRSPTPSSSPRPSSSPPPSSSSPPPSSSMPPPPPPLPPK"
60515 print"PRSSDSSSSSNSSSS2RPSSSSPR3SSSSSMPPPPPKPPK"
60516 print"5HSSDSSSSNSSSSSSTSSSSSSTSSSSSSMPPPPKPPS"
60517 print"GRSSDSSPPPPSSSSSSSSSSSSSSSSSSSSSPPPPKPPQ"
60518 print"RHSSDSSGGGRSPPPPPPPPPPPPPSPPPPSSRPPPKPPK"
60519 print"GRSSDSSPPPTSRPRRPPPRRPPRSPAARSSIIIIKPPK"
60520 print"RHSSDSSTTTTSIIIIQIIIIQIIMPPPRSSIPSSSSPK"
60521 print"RPP1DPPPPPPPSSSSSSSSSSSSSSSSMPRRSSPPSSSS0K"
60522 print"AAAAAAAAAAAASSSSSSSSSSSSSSSAQAAIIIQSSSAAB"
60580 return
ready.
```

Listing »Spring-Vogel« (Schluß)

1520-Hardcopy mit dem VC 20

Dieses Programm ermöglicht es allen VC 20-Besitzern, Grafik-Hardcopys mit dem Printer/Plotter VC 1520 von Commodore anzufertigen.

Das Programm ist einfach mit »LOAD« und den gerätespezifischen Parametern zu laden. Da es sich um ein Basicprogramm handelt, wird keine Sekundäradresse benötigt.

Mit RUN wird das Basicprogramm gestartet, welches die eigentliche Maschinensprachroutine in Form von DATA-Statements enthält. Das Programm fragt nun nach der Zieladresse, ab welcher die Routine abgelegt werden soll. Ist dabei der angegebene Wert gleich Null, so werden 388 Byte vom Basic-Pointer ($55/56) abgezogen und das Programm dahinter abgelegt. Wird ein anderer Wert eingegeben, so wird das Programm ab dieser Adresse gespeichert und der Pointer bleibt unverändert. Nach dieser Eingabe braucht man sich, da das Basicprogramm über einen Relocator und Prüfsummenabfrage verfügt, nicht mehr um das weitere Ablesen des Maschinenspracheprogramms zu kümmern. Sollte sich ein Prüfsummenfehler ergeben, so wird dieser angezeigt und der DATA-Block mit dem Maschinenprogramm muß auf Tippfehler hin untersucht werden.

Am Ende des Ladevorganges wird die Position der Routine und die Startadresse für den SYS-Befehl angezeigt. Jetzt kann »HARDCOPY 60« mit SYS 0 oder der ausgegebenen Absolutadresse gestartet werden.

Folgende Einschränkungen sind zu beachten:

(a) Falls ein File mit der logischen File-Nummer 127 geöffnet wurde, so ist dieses vor dem Start von »HARDCOPY 60« wieder zu schließen.

(b) Der Startbefehl darf nicht im »Direktmodus« stehen.

Basic-Lader für »Hardcopy 60«

```
100 rem--------------------
105 rem"  HARDCOPY  60   "
110 rem--------------------
115 rem"createt 04'84 by"
120 rem"                "
125 rem"W.W.Wirth       "
130 rem"Th.Heuss Ring 20"
135 rem"6556  Woellstein"
140 rem--------------------
145 rem"(c) by          "
150 rem"W.W.Wirth & SDG "
155 rem--------------------
160 rem
165 rem
170 clr:poke36879,14
175 print"■■■  Ladeprogramm fuer"
180 print"■  ■*  HARDCOPY 60  *"
185 print"■■  Wo soll das"
190 print"■  Machinenprogramm"
195 print"■  abgelegt werden ?"
200 input"■    0■■■";za
205 gosub415:ifzagoto230
210 za=fndp(55)-389
215 poke55,fnlb(za)
220 poke56,fnhb(za)
225 clr:gosub415:za=fndp(55)
230 of=za-28672
235 print"■  Das Prg. wird jetzt"
240 print"■  abgelegt.■"
245 fori=0to388:readby$
250 print" "i"■"
255 by=16*fnb4(asc(by$))+fnb4(asc(right$(by$,1)))
260 pokeza+i,by:cs=cs+by
265 next
270 ifcs=37822goto300
275 print"■■ Checksummenfehler !"
280 print"■ Bitte Machinenpro-"
285 print"■ gramm-Datablock"
290 print"■ ueberpruefen !"
295 end
300 print"■ ■ Das Programm wird"
305 print" ■ jetzt abgeaendert.■"
310 fori=0to13
315 readrp:ap=rp+za
320 print" "ap"■"
325 j=fndp(ap)+of
330 pokeap,fnlb(j)
335 pokeap+1,fnhb(j)
340 next
345 poke0,76:j=28869+of
350 poke1,fnlb(j)
355 poke2,fnhb(j)
360 print"■■ Das Programm ist"
365 print"■ startbereit."
370 print"■ Es belegt RAM von■"
375 printza" bis"za+388"."
380 print"■ Es wird mit"
385 print"■ 'SYS 0' oder"
390 print"■ 'SYS"j"▒'"
395 print"■ gestartet.":end
400 rem--------------------
405 rem"      UP         "
410 rem--------------------
415 deffnhb(x)=int(x/256)
420 deffnlb(x)=x-fnhb(x)*256
425 deffnb4(x)=x-48+(x>57)*7
430 deffndp(x)=peek(x)+256*peek(x+1)
435 return
440 rem--------------------
445 rem"      ML-PRG     "
450 rem--------------------
455 data48,a2,7f,86,13,20,15,e1,68,20,09
460 datae1,ae,04,02,ac,05,02,e0,01,ad,08
465 data02,60,a9,4d,20,00,70,10,04,b0,01
470 data88,ca,20,38,70,a9,44,20,00,70,30
475 data06,e8,d0,08,c8,b0,05,b0,01,88,ca
480 dataca,98,20,cd,dd,20,3f,cb,a9,2d,20
485 data09,e1,ae,06,02,ad,07,02,20,cd,dd
490 data4c,b5,cb,a2,00,86,fc,a2,03,ac,02
495 data02,c0,10,d0,01,e8,0a,26,fc,ca,d0
500 datafa,85,fb,a5,fc,6d,03,02,30,06,c9
505 data20,90,02,69,5f,85,fc,ac,09,02,b1
510 datafb,a2,08,86,fb,ae,08,02,30,02,0a
515 dataa2,4a,48,90,03,20,18,70,20,0c,70
520 data30,07,e8,e8,d0,08,c8,b0,05,b0,01
525 data88,ca,ca,8e,04,02,8c,05,02,68,c6
530 datafb,d0,d6,60,ee,06,02,d0,03,ee,07
535 data02,60,a0,00,ad,a0,01,a9,7f,a2,06
540 data20,50,fe,a9,00,85,b7,4c,be,e1,20
545 dataa6,d3,a9,08,8d,02,02,ad,02,90,08
550 data29,7f,8d,00,02,ad,03,90,4a,90,03
555 data0e,02,02,29,3f,8d,01,02,ad,05,90
560 dataaa,08,4a,4a,29,1c,28,30,02,09,80
565 data28,10,02,09,02,85,fe,8a,29,08,08
570 data8a,0a,0a,29,1c,28,d0,02,09,80,8d
575 data03,02,a9,00,85,fd,a2,03,9d,04,02
580 dataca,10,fa,20,72,71,20,b5,70,a0,00
585 data8c,09,02,4e,08,02,a0,00,8c,0a,02
590 dataac,0a,02,b1,fd,20,50,70,ac,0a,02
595 datac8,cc,00,02,90,ec,6e,08,02,20,a9
600 data70,88,8c,0a,02,ac,0a,02,b1,fd,20
605 data50,70,ac,0a,02,88,10,ef,20,a9,70
610 dataac,09,02,c8,cc,02,02,90,c0,18,a5
615 datafd,6d,00,02,85,fd,90,02,e6,fe,ce
620 data01,02,d0,ad,20,80,71,20,b2,70,a9
625 data0d,20,00,70,20,d7,ca,20,b5,cb,a9
630 data7f,4c,c9,e1
635 rem--------------------
640 rem" Position        "
645 rem" der 'JSR'       "
650 rem" im ML-PRG       "
655 rem--------------------
660 data27,41,376,141,138,36,303,330
665 data318,339,371,282,279,368

ready.
```

Bild 2. Auch hochauflösende Grafik wird problemlos gedruckt.

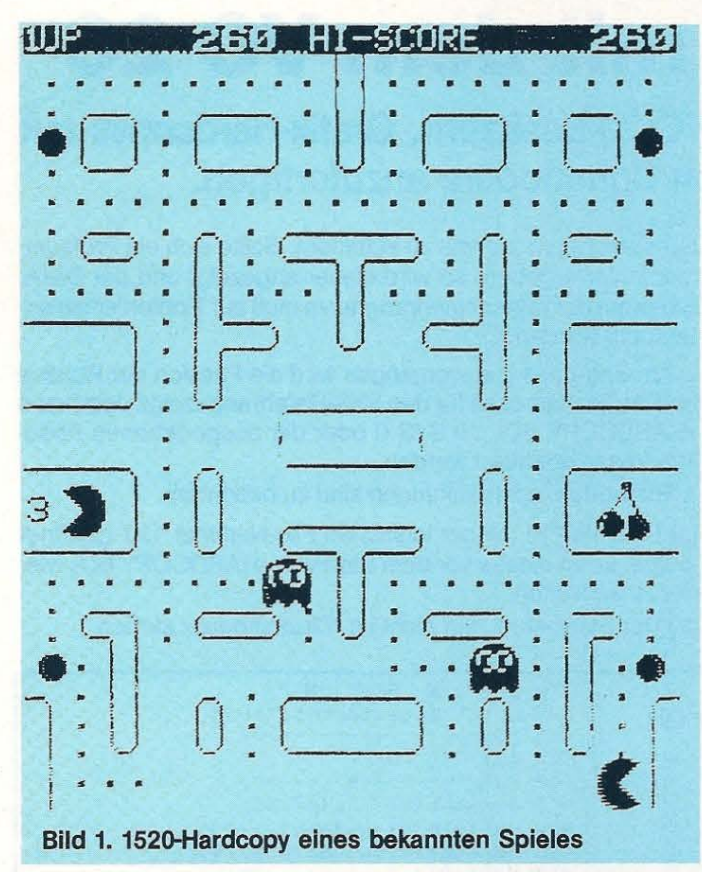


Bild 1. 1520-Hardcopy eines bekannten Spieles

Das Maschinenprogramm liest beim Aufruf die aktuellen Daten aus den Registern des Video-Interface-Controller (VIC) und berechnet daraus folgende Parameter:

Position	Bestimmung
$0200	Bildschirmbreite
$0201	Bildschirmlänge
$0202	Länge eines Zeichens (8x8 oder 8x16)
$0203	HI-Byte der Position des 1. Bytes des 1. Zeichens im Charakter-RAM/-ROM
$0204/5	Absolute MOVE-/DRAW-Position in horizontaler Richtung
$0206/7	desgleichen in vertikaler Richtung
$0208	Vorwärts-/Rückwärts-Flag
$0209	Nummer eines Bytes im Zeichengenerator bezüglich eines Bildschirmzeichens
$020a	Hilfszeiger auf momentane Bildschirmposition
$fb/c	Zeiger auf Byte im Zeichengenerator
$fd/e	desgleichen für Video-RAM

Das Programm liest von links nach rechts die Bildschirm-Codes aus dem Video-RAM und berechnet unter Zuhilfenahme der oben genannten Parameter die Position der Bitmuster im Zeichengenerator. Ist ein Bit gesetzt, so wird seine Position in den MOVE-Befehl des Plotters umgesetzt und dort ein Strich der Länge 1 gezogen.

Ein Video-Punkt entspricht vier Plotter-Punkten woraus sich eine Auflösung von 30 Zeichen pro Plotterzeile ergibt. Dieser 2x2-Punkt hätte einen zweiten Plotvorgang nötig gemacht. Tatsächlich wurde aber die Dauer eines Plot-Vorgangs halbiert indem der Wagenrücklauf des Stiftes mitbenutzt wird. Eine weitere Zeitoptimierung wird dadurch erreicht, indem keine »Leer-Plot«-Befehle ausgegeben werden. Der Stift wird also nur dann bewegt, wenn ein Bit auch gesetzt ist.

Das dauernde »Ticken« des Stiftes läßt sich leider nicht vermeiden, da pro Bit eine neue Positionierung des Stiftes nötig ist (sonst hätte ein Punkt die Größe 3x2).

»Hardcopy 60« kann sowohl für normale Texte, als auch für Grafik-Bildschirme eingesetzt werden.

(Wolfgang W. Wirth/ev)

Druckfehlerteufelchen

Folgende Fehler sind in den Ausgaben 7 und 8 auf dem Konto von unserem Teufelchen gutgeschrieben worden.

Komfortables Treiberprogramm für Centronics-Drucker, 7/84, Seite 110

Ein Leserbrief hat ergeben, daß der Drucker NEC 8032 im Bitmustermodus das niederwertige Bit nicht wie der Epson-Drucker unten sondern oben druckt. Damit stehen alle ausgedruckten Bildschirmzeichen bei Verwendung meines Treiberprogramms auf dem Kopf. Bei diesem Drucker müssen deshalb im Programm 2 Byte geändert werden. In Zeile 260 das 2. Datum in 128 und in Zeile 264 das 3. Datum in 70.

(Helmut Eyssele)

Hardcopy mit dem VC 1520, 7/84, Seite 108

In dem einleitenden Text sind zwei Fehler vorhanden, ein Druck- und ein Denkfehler. Der Druckfehler ist in der POKE-Zeile. Da muß das + durch ein * ersetzt werden. Der Denkfehler ist, daß dies eigentlich überhaupt nicht notwendig ist. Als ich Ihnen das HC 1520-Programm zuschickte, war es ein Teilprogramm in der Pic-Show 1520 und nur für diese Pic-Show war die POKE-Zeile notwendig. Ein Fehler ist auch mir unterlaufen. Die letzte Zeile wird nicht geplottet. Dieser Fehler ist aber leicht zu korrigieren. Einfach die Zahl 7680 in Zeile 330 in die Zahl 8000 verwandeln.

(Jörg Wichmann)

Zwei Einzeiler, 7/84, Seite 135

Der zweite Einzeiler muß korrekt lauten:

```
x$=" ":fori=1to4:x0=x/16:x-
=
x-int(x0)*16:x$=chr$(48+x-
(x(9)*7)+x$:x=int(x0):next
```

Centronics-Schnittstellen, 7/84, Seite 13

In Zeile 110 der Hardcopyroutine für das Görlitz-Interface muß vor dem V unbedingt ein SPACE in den Anführungszeichen eingefügt werden. Die korrekte Zeile lautet:

```
110 PRINT#1,CHR$(27)"V".
```

Vollautomatisches Blumengießen, 7/84, Seite 82

1. Der Minuspol vom Netzteil muß mit der Masse (GND) vom User-Port verbunden sein, da die Ansteuerung von T1 sonst nicht klappt.
2. Der Pfeil im Schaltzeichen von T1 ist umzudrehen.
3. C1 muß aufgrund seiner Größe ein Elco sein. Im Schaltzeichen muß daher ein + am Pluspol des Gleichrichters eingezeichnet sein.
4. In einer Zeitschrift für Software-Anwender ist der Hinweis, daß 220 Volt, vor allem im Zusammenhang mit Wasser, auch für Hardware-Bastler tödlich sein können, sicherlich angebracht.

Die Software-Spezialisten danken dem Hardware-Profi und Leser Michael Scharf für diese Hinweise.

Was ist Comal?, 8/84, Seite 41

Die angebene Adresse der Firma INSTRUTEK in Dänemark ist leider nicht ganz richtig. Genauer, die Adresse stimmt schon, nur kann man dort die Sprache Comal nicht umsonst beziehen. Instruktek bietet nämlich nur die Version 2.0 von Comal für die großen CBMs an, und die kostet um die 600 Mark.

Die von uns besprochene Version 0.14 gibt es als sogenannte Public Domain Software gegen Verpackungs- und Versandkosten bei:
Interpool
c/o Prof. Burkard Leuschner
Wiesengrund 6
7487 Gammetingen-Bronnen
Tel: 07574/3728

Steuerzeichen

Unser Drucker beherrscht immer noch nicht alle Commodore-Steuerzeichen. So ist das Zeichen »_« durch Pfeil links, »^« durch Pfeil nach oben, »/« (revers) durch das reverse Pfundzeichen und »§« durch den Klammeraffen (@) zu ersetzen.

Pascal, 7/84, Seite 44

Die Version Pascal 64 von Data Becker ist seit einem halben Jahr nicht mehr erhältlich und durch Pascal 64 Version 3.0 vollständig ersetzt worden. Der Umtausch ist für 50 Mark (mit neuem Handbuch) möglich.

Der Super-Sprite-Editor

Zirka fünfzig verschiedene Sprite-Editorprogramme erreichten uns in der Redaktion. Das beste Programm haben wir für Sie ausgewählt.

Der Hauptbildschirm besteht aus einem Anzeigefeld (Spalte 0 bis 12) und einem Arbeitsfeld (rechts eingerahmt). Im Anzeigefeld stehen das zu bearbeitende Sprite und Hinweise auf verfügbare Kommandos, im Arbeitsfeld wird das Sprite erstellt. Die Zahlen im Rahmen des Arbeitsfeldes ergeben sich aus dem Format der Sprites im Speicher.

Das Menü

Das Menü (=Command List) wird mit »Space« ins Anzeigefeld gerufen.

0 bis 7
Es werden bis zu sieben Sprites gleichzeitig im Spritefeld dargestellt. Die Tasten 0 bis 7 fungieren als On/Off-Schalter.

I
Das jeweils zuletzt angeschaltete Sprite wird nacheinander in Y-, XY- und X-Richtung vergrößert.

M
Schaltet für das zuletzt bearbeitete Sprite den Multicolormodus ein.

C
Verzweigt in die Routine für die Farbwahl.

R
Reproduktion des zuletzt bearbeiteten Sprites ins Arbeitsfeld. Kann mit beliebiger Taste abgebrochen werden.

F1
Schaltet in den Arbeitsmodus (siehe unten).

F3
Ändert die Hintergrundfarbe.

F7
Schiebt das Sprite im Speicher um eine Zeile tiefer. Es kann passieren, daß Punkte aus dem Sprite davor ins Bild verschoben werden.

F8
Wie F7, schiebt das Sprite hoch.

H
Verzweigt in die Handle-Routine, mit der Sprites auf dem Bildschirm verschoben werden können.

$
Zeigt das Diskettendirectory an. Kann mit F 1 abgebrochen werden.

@
Ermöglicht Disk-Befehle zu senden, beziehungsweise nur mit RETURN den Fehlerkanal abzufragen.

S
Gibt die Dezimalwerte des zuletzt bearbeiteten Sprite im Anzeigefeld aus. Am Kopf der Tabelle steht die Anfangsadresse des Sprites im Speicher (dezimal).

P
Ausgabe auf Drucker. Zusätzliche Angabe eines 20 Zeichen langen Namens möglich. Geräteadresse = 4

F
Verzweigt ins Floppymenü. Dieser Programmteil erklärt sich weitgehend selbst. Beim ersten Einsprung in die Routine sollte ein Name definiert werden, sonst wird das File (sequentiell) unter dem Namen »Data« geSAVEd. Später kann die Position »Filename« mit RETURN übergangen werden und der alte Name wird verwendet. Unter einem Namen können beliebig viele Sprites abgelegt werden. Geräteadresse = 8

D
Alle acht Sprites werden in Datazeilen umgewandelt.

K
Das Programm wird gelöscht. Datazeilen bleiben bestehen und können an andere Programme angehängt werden. Geduld, die Sache dauert!

Der Arbeitsmodus

'*' beziehungsweise '@'
Diese Tasten zeichnen Punkte (Linien) in horizontaler beziehungsweise vertikaler Richtung.

'=' beziehungsweise ';'
Wie oben, nur werden Punkte gelöscht. Die Belegung dieser Tasten ist an die Cursortasten angelehnt. Gleichzeitig mit SHIFT ist die Bewegungsrichtung umgekehrt.
Außerdem definiert: RETURN, HOME, CLEAR, F1

Aus fast jeder Routine kann mit F1 ausgestiegen werden. Auf Fragen wird mit Y(es) oder RETURN und N(o) oder beliebiger Taste geantwortet. Nicht definierte Tasten bewirken nichts. Fehleingaben sind abgefangen.

Man sollte das Programm nicht mit STOP/RESTORE beenden. Der Bildschirminhalt ist sonst nicht mehr lesbar. Nach QUIT oder STOP kann allerdings problemlos ohne Verlust der Sprites neu gestartet werden.

Für »Tippfaule«: Sämtliche Remzeilen sind optional, da sie nicht angesprungen werden, und dienen nur der Orientierung im Listing. Man kann sie also auch später einfügen, oder ganz weglassen.

Und wenn's dann endlich drin ist (wunde Finger gehören nunmal dazu): Viel Spaß beim Malen!

Erweiterungen

1) Der Bildschirm ist nach 33792 verlegt, um für acht Sprites Platz zu schaffen. Dementsprechend liegen die Sprites auch in Block 3, nämlich ab Adresse 32768.
2) Die Tastenbelegung der »Work«-Routine ist der Verwendung von Cursortasten angeglichen. Links vertikale, rechts horizontale Bewegungen.
3) Es stehen drei Routinen für Diskettenverwaltung zur Verfügung:
a) Anzeige des Directory
b) Senden von Disk-Commands
c) Floppy Controller
4) Die »Floppy«-Routine ist so gestaltet, daß ein relativ bequemes Umsortieren der Sprites auf Diskette von einem File ins andere möglich wird.

Das heißt: Viele Stellen, an denen bei Irrtum noch eine »Umkehr« möglich ist; immer wieder wird gezeigt, was gerade passiert. Bei Eingabe des Filenamens kann jetzt mit DELETE korrigiert werden. Bei fehlender Namensangabe heißt das File »Data« — kein unlöschbares »,« mehr.

Wenn man in letzter Minute noch aussteigt und doch kein einziges Sprite abgespeichert, entsteht nun kein Geisterfile mehr, das zwar einen Eintrag im Directory hat, aber keine Daten enthält und das Programm bei nächster Gelegenheit böswillig zum Absturz bringt.
5) Diverse Kleinigkeiten.

(A. Kölbach/rg)

```
0 REM ................................
1 REM .                              .
2 REM .     S P R I T E A I D +      .
3 REM .                              .
4 REM . WRITTEN BY ANDREAS KOELBACH .
5 REM .              STADTWALDSTR. 5 .
6 REM .              3550 MARBURG/L. .
7 REM .                              .
8 REM ................................
9 REM
10 V=53248:IFPEEK(53280)<>246THENPOKE532
80,6:GOSUB345
11 CD$="▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒"
12 F=55296:C=33792:SN=32768:A=0:O$="
"
13 FORI=0TO7:POKE53287+I,7:POKEC+1016+I,
I:NEXT
14 POKE650,128:POKEV+28,0:POKEV+23,0:POK
EV+29,0
15 DEFFNA(A)=A+X+Y*40
16 GOSUB24
17 POKEV+33,6:PRINT"▒▒.SPRITEAID+.▒"
18 PRINT"                ▒      8       6
      4 ▒"
19 FORI=1TO20STEP2
20 PRINT"               ▒ ▒..............
..........▒ ▒"
21 PRINT"               ▒-▒..............
..........▒-▒":NEXT
22 PRINT"               ▒ ▒..............
..........▒ ▒"
23 PRINT"               ▒      8       6
      4 ▒":GOSUB172:GOTO68
24 POKEV+21,0:FORI=0TO7:S(I)=1:SN$(I)=".
":NEXT:RETURN
25 REM******* CALCULATE DOT  ***********
26 CA=X-14:CB=Y-2:H=INT(CA/8):BY=SN+3*CB
+H:BI=2^(7-CA+H*8)
27 IFZL=46THENPOKEBY,PEEK(BY)AND255-BI:R
ETURN
28 POKEBY,PEEK(BY)ORBI:RETURN
29 REM ******** GET SUBROUTINE *********
30 POKE204,0:POKE198,0:WAIT198,1:GETA$:A
=ASC(A$)
31 IFPEEK(207)THEN31
32 POKE204,1:RETURN
33 POKE198,0:WAIT198,1:GETA$:A=ASC(A$):R
ETURN
34 REM ********* REPRODUCTION **********
35 PRINT"▒"TAB(13)"▒- REPRO -▒":CA=0:BI=
0
36 FORBY=SNTOSN+62:B=PEEK(BY):FORI=7TO0S
TEP-1:CA=CA+1
37 IFBAND2^ITHENPOKEC+93+CA,42:GOTO39
38 POKEC+93+CA,46:GETA$:IFA$<>""THENI=0:
BY=SN+62
39 NEXTI:BI=BI+1:IFBI=3THENBI=0:CA=CA+16
40 NEXTBY:RETURN
41 REM ********** MULTIMATRIX ***********
42 A=0:GOTO44
43 A=14
44 IFMF(S)=0THENRETURN
45 FORI=55387TO56213STEP40:FORJ=1TO24STE
P4:POKEI+J,A:POKEI+J+1,A:NEXTJ,I:RETURN
46 REM *********** CLEAR SPRITE *********
47 POKEFNA(C),ZL:POKEFNA(F),FL
48 PRINTLEFT$(CD$,14)"▒SURE ? ▒";:GOSUB3
0:IFA<>89ANDA<>13THEN52
49 MF(S)=1:GOSUB183
50 PRINT"▒▒▒";:FORI=1TO21:PRINTTAB(14)".
.......................":NEXT
51 FORI=SNTOSN+62:POKEI,0:NEXT
52 PRINTLEFT$(CD$,14)"            ":GOSUB43:GO
TO68
53 REM********* SET CURSOR *************
54 POKEFNA(C),ZL:POKEFNA(F),FL
55 IFX+R<38ANDX-L>13ANDY-U>1ANDY+D<23THE
N60
56 IFLTHENX=38:GOTO60
57 IFUTHENY=23:GOTO60
58 IFDTHENY=1:GOTO60
59 IFRTHENX=13
60 X=X+R-L:Y=Y+D-U
61 ZL=PEEK(FNA(C)):FL=PEEK(FNA(F))
62 POKEFNA(C),43:POKEFNA(F),1
63 R=0:L=0:D=0:U=0:RETURN
64 REM ********** SLIP SPRITE **********
65 FORI=SN+62TOSNSTEP-1:POKEI,PEEK(I-3):
NEXT:RETURN
66 FORI=SNTOSN+62:POKEI,PEEK(I+3):NEXT:R
ETURN
67 REM ******** MAIN MENU **************
68 PRINT"▒"TAB(13)"                    "
69 PRINTLEFT$(CD$,13)
70 PRINT"                    "
71 PRINT"                    "
72 PRINT"                    "
73 PRINT"                    "
74 PRINT"▒ SPACE ▒ FOR"
75 PRINT"COMMAND LIST "
76 PRINT"                    "
77 PRINT"                    "
78 PRINT"                    "
79 PRINT"                    ":PRINT"
"
80 PRINTLEFT$(CD$,24)"INPUT NO.? ";
81 GOSUB30
82 IFA>47ANDA<56THEN105:REM ON/OFF
83 IFA=67THEN192:REM COLOR
84 IFA=73THEN111:REM INCR.
85 IFA=77THENGOSUB183:GOTO80:REM MULTI
86 IFA=82THENGOSUB35:GOTO68:REM REPRO
87 IFA=81THEN154:REM EXIT
88 IFA=133THENGOSUB42:GOTO242:REM WORK
89 IFA=136THENGOSUB65:GOTO80:REM SLIP
90 IFA=140THENGOSUB66:GOTO80:REM SLIP
91 IFA=134THENGOSUB103:REM BACK COLOR
92 IFA=72THENGOSUB24:GOTO411:REM HANDLE
93 IFA=83THEN145:REM SCREENDAT
94 IFA=80THEN221:REM PRINTER
95 IFA=70THENGOSUB24:GOTO276:REM FLOPPY
96 IFA=68THEN362:REM DATALINE
97 IFA=75THENGOSUB24:GOTO439:REM KILL
98 IFA=36THEN120:REM $
99 IFA=64THEN374:REM @
100 IFA=32THEN387:REM COMMAND LIST
101 GOTO80
102 REM ******** BACKGROUND ************
*
103 POKEV+33,PEEK(V+33)+1AND15:RETURN
104 REM ******** SPRITE ON/OFF *********
*
105 S=A-48:SN=32768+S*64
106 IFS(S)=0THENS(S)=1:SN$(S)=".":POKEV+
21,PEEK(V+21)AND255-2^S:GOSUB172:GOTO80
107 S(S)=0:SN$(S)=RIGHT$(STR$(S),1):GOSU
B172:POKEV+2*S,56:POKEV+1+2*S,92
108 POKEV+21,PEEK(V+21)OR2^S:IFFI(S)=0TH
EN80
109 FI(S)=FI(S)-1
```

Listing »Spriteaid+«

```
110 REM********* INCREASED ************
111 GOSUB162:IFFI(S)THEN113
112 FI(S)=1:POKEV+23,PEEK(V+23)OR2^S:GOT
080
113 IFFI(S)>1THEN115
114 FI(S)=2:POKEV+29,PEEK(V+29)OR2^S:GOT
080
115 IFFI(S)>2THEN117
116 FI(S)=3:POKEV+23,PEEK(V+23)AND255-2^
S:GOTO80
117 POKEV+29,PEEK(V+29)AND255-2^S
118 GOSUB172:FI(S)=0:GOTO80
119 REM ********* DIRECTORY ************
120 GOSUB24:PRINT"▒▒.DIRECTORY.▒▒":I=0
121 OPEN2,8,15:OPEN1,8,0,"$"
122 GET#1,A$,B$
123 GET#1,A$,B$
124 GET#1,A$,B$:I=I+1
125 B=0:IFA$<>""THENB=ASC(A$)
126 IFB$<>""THENB=B+ASC(B$)*256
127 PRINTMID$(STR$(B),2);TAB(5);
128 GET#1,B$:IFSTTHEN138
129 IFB$<>CHR$(34)THEN128
130 GET#1,B$:IFB$<>CHR$(34)THENPRINTB$;:
GOTO130
131 GET#1,B$:IFB$=CHR$(32)THEN131
132 PRINTTAB(21);:C$=""
133 C$=C$+B$:GET#1,B$:IFB$<>""THEN133
134 PRINT"  "LEFT$(C$,5)
135 GETT$:IFT$="▒"THEN142
136 IFI=20THEN139
137 IFST=0THEN123
138 PRINT"▒ BLOCKS FREE ▒":CLOSE1:CLOSE2
:GOSUB141:GOTO16
139 PRINT"▒      PRESS ANY KEY ...":GOSUB1
41
140 PRINT"▒▒.DIRECTORY.▒▒":I=0:GOTO123
141 GETT$:IFT$=""THEN141
142 IFT$="▒"THENCLOSE1:CLOSE2:GOTO16
143 RETURN
144 REM ******* GIVE OUT DECIMAL *******
145 GOSUB24:GOSUB156:PRINT"▒▒ ADR.";SN;"
▒":FORI=SNTOSN+62STEP3
146 A1$=STR$(PEEK(I)):A2$=STR$(PEEK(I+1)
):A3$=STR$(PEEK(I+2))
147 A1$=LEFT$(O$,4-LEN(A1$))+MID$(A1$,1,
4)
148 A2$=LEFT$(O$,4-LEN(A2$))+MID$(A2$,1,
4)
149 A3$=LEFT$(O$,4-LEN(A3$))+MID$(A3$,1,
4)
150 PRINT"▒";A1$;A2$;A3$:NEXT
151 GOSUB159:POKE198,0:WAIT198,1
152 GOSUB156:GOSUB172:GOTO68
153 REM ************* EXIT *************
154 PRINT"▒▒";:END
155 REM ****** CLEAR DISPLAY AREA ******
156 PRINT"▒"
157 FORI=1TO23:PRINT"              ":NEXT
:PRINT"▒";
158 REM ****** SCREEN LINE MSB *********
159 FORI=0TO6:POKE217+I,132:POKE230+I,13
4:NEXT
160 FORI=0TO5:POKE224+I,133:POKE237+I,13
5:NEXT:RETURN
161 REM********************************
162 PRINT"▒▒▒▒▒███ ┌────────┐"
163 PRINT"███        │"
164 PRINT"███        │"
165 PRINT"███        │"
166 PRINT"███        │"
167 PRINT"███        │"
168 PRINT"███        │"
169 PRINT"███└───────┘"
170 RETURN
171 REM**********************************
172 PRINT"▒▒▒  ";:FORI=0TO7:IFI=STHENPRI
NT"▒";
173 PRINTSN$(I)"▒";:NEXT:PRINT
174 PRINT"▒ ┌─────┐ "
175 PRINT"  │     │   "
176 PRINT"  │     │   "
177 PRINT"  │     │   "
178 PRINT"  └─────┘   "
179 PRINT"            "
180 PRINT"            "
181 PRINT"            ":RETURN
182 REM ********** MULTI MODE **********
183 IFMF(S)THEN185
184 POKEV+28,PEEK(V+28)OR2^S:MF(S)=1:GOT
0186
185 POKEV+28,PEEK(V+28)AND255-2^S:MF(S)=
0
186 PRINTLEFT$(CD$,13);
187 PRINT"MLT:";
188 FORI=0TO7:IFMF(I)THENPRINTRIGHT$(STR
$(I),1);:GOTO190
189 PRINT".";
190 NEXT:PRINT:RETURN
191 REM ********** SET COLOR ***********
192 A=PEEK(V+37):C$(0)=STR$(A-240)
193 A=PEEK(V+38):C$(1)=STR$(A-240)
194 A=PEEK(V+39+S):C$(2)=STR$(A-240)
195 PRINTLEFT$(CD$,16);
196 PRINT"COLORS  REG. "
197 PRINT"             "
198 PRINT"          37 "
199 PRINT"             "
200 PRINT"          38 "
201 PRINT"             "
202 PRINT"          ▒"STR$(39+S)" ▒ "
203 PRINT"             "
204 PRINT"             "
205 PRINTLEFT$(CD$,18)" "C$(0)" "
206 PRINT"▒▒"C$(1)" ":PRINT"▒▒"C$(2)" "
207 PRINTLEFT$(CD$,16)
208 I=0:GOSUB212:POKEV+37,B
209 I=1:GOSUB212:POKEV+38,B
210 I=2:GOSUB212:POKEV+39+S,B
211 GOTO68
212 PRINT"▒  ";:B$=""
213 GOSUB30:IFA=13THEN217
214 IFA<48ORA>57THEN213
215 B$=B$+A$:PRINTA$;:IFLEN(B$)=2THEN218
216 GOTO213
217 IFB$=""THENPRINT:GOTO219
218 C$(I)=B$:PRINT" "
219 B=VAL(LEFT$(C$(I),3)):RETURN
220 REM ********** PRINTER OUT *********
221 PRINTLEFT$(CD$,25);
222 PRINT"NAME (20CHR.) ?
     ▒":PRINTTAB(16);:B$=""
223 GOSUB30:GOTO225
224 GOSUB33
225 IFA=20ANDB$<>""THENB$=LEFT$(B$,LEN(B
$)-1):PRINTCHR$(20);:GOTO224
226 IFA=133THEN16
227 IFA=13THEN231
228 IFA=<32ORA>127THEN224
229 B$=B$+A$:IFLEN(B$)>20THEN231
```

Listing »Spriteaid+« (Fortsetzung)



```
230 PRINTA$" ";:GOTO224
231 PRINTLEFT$(CD$,11)TAB(17)" PRINTER
 ON ?     ";
232 GOSUB30
233 IFA=78THEN221
234 IFA=133THEN16
235 IFA<>13ANDA<>89THEN231
236 OPEN1,4:CMD1:PRINTB$
237 FORI=SNTOSN+62STEP3:
238 PRINTPEEK(I);PEEK(I+1);PEEK(I+2)
239 NEXT:PRINT
240 CLOSE1:GOTO16
241 REM ********* WORK ROUTINE *********
242 PRINT" "TAB(14)" - WORK - "
243 PRINTLEFT$(CD$,15);
244 PRINT"                   "
245 PRINT"DOT(*)='*'         "
246 PRINT"       '@'         "
247 PRINT"SPC(.)='='         "
248 PRINT"       ';'         "
249 PRINT"                   "
250 PRINT"MENU     ='F1'    "
251 PRINT"                   "
252 PRINT"  USE'CRSR'        "
253 PRINT"   TO MOVE !    ";
254 X=0:Y=0:D=1:R=14:ZL=46:FL=1
255 GOSUB54
256 POKE198,0:WAIT198,1:GETA$:A=ASC(A$)
257 IFA=145THENU=1:GOTO255
258 IFA=157THENL=1:GOTO255
259 IFA=17THEND=1:GOTO255
260 IFA=29THENR=1:GOTO255
261 IFPEEK(654)THEN269
262 IFA=42THENR=1:ZL=42:GOSUB26:GOTO255
263 IFA=64THEND=1:ZL=42:GOSUB26:GOTO255
264 IFA=59THEND=1:ZL=46:GOSUB26:GOTO255
265 IFA=61THENR=1:ZL=46:GOSUB26:GOTO255
266 IFA=13THENL=X-14:GOTO255
267 IFA=133THENPOKEFNA(C),ZL:POKEFNA(F),
FL:GOSUB43:GOTO68
268 IFA=19THENL=X-14:U=Y-2:GOTO255
269 IFA=192THENL=1:ZL=42:GOSUB26:GOTO255
270 IFA=93THENU=1:ZL=46:GOSUB26:GOTO255
271 IFA=186THENU=1:ZL=42:GOSUB26:GOTO255
272 IFA=61THENL=1:ZL=46:GOSUB26:GOTO255
273 IFA=147THEN47
274 GOTO256
275 REM ******** FLOPPY ****************
276 POKEV+21,0:PRINT" "LEFT$(CD$,25)"
     - 'F1'=EXIT - ";
277 FORI=0TO7:SP(I)=0:NEXT
278 PRINT" .FLOPPY DISK CONTROLLER. "
279 PRINT" READ OR WRITE ?"
280 GOSUB33
281 IFA=133THEN16
282 IFA$="R"THENM=0:B$=" READ ":C$="":B=
7:GOTO285
283 IFA$="W"THENM=1:B$=" WRITE ":C$="S":
B=0:GOTO285
284 GOTO278
285 PRINT" MODE :          ":PRINTTAB(8)
B$:IFM=0THENC$="REPLACE"
286 PRINT" INPUT SPRITE"C$" YOU WANT TO
"B$:B$="":PRINT" > ";
287 GOSUB33
288 IFA=133THEN276
289 IFA=13THEN292
290 IFA<48ORA>55THEN287
291 A=VAL(A$):PRINTA;:SP(A)=1:B=B+1:IFB<
8THEN287
```

Listing »Spriteaid+« (Fortsetzung)

```
292 PRINT" < "
293 F$="":PRINT"FILENAME : ";
294 GOSUB33:IFA=20ANDLEN(F$)>0THENF$=LEF
T$(F$,LEN(F$)-1):PRINTA$:
295 IFA=13THEN300
296 IFA<32ORA>127THEN294
297 PRINTA$;:F$=F$+A$
298 IFLEN(F$)>16THENPRINT" ";:GOTO300
299 GOTO294
300 IFF$<>""THENFI$=F$:GOTO302
301 PRINTFI$;:IFFI$=""THENFI$="DATA"
302 FORI=0TO7:FI(I)=0:NEXT:POKEV+23,0:PO
KEV+29,0:X=0:Y=0:IFMTHEN321
303 REM ********** READ ***************
304 GOSUB339:GOSUB33:IFA<>13ANDA<>89THEN
276
305 B$=" >>> READING <<< ":GOSUB343:NO=0
306 REM ********** INPUT **************
307 OPEN15,8,15:OPEN2,8,2,FI$+",S,R"
308 INPUT#15,A,B$:IFATHENGOSUB318:GOTO27
6
309 PRINTTAB(3)" NO.:"NO:NO=NO+1
310 FORI=ADTOAD+62:INPUT#2,B
311 POKEI,B:NEXTI
312 IFST=64THENB$=" END OF DATA ! ":GO
SUB343:GOSUB33:GOTO316
313 B$="  TAKE OVER ?   ":GOSUB343
314 GOSUB33:IFA=133THEN316
315 IFA<>13ANDA<>89THENB$=" >>> READING
<<< ":GOSUB343:GOTO309
316 POKEV+21,0:CLOSE2:CLOSE15:GOTO276
317 REM ********** ERRORS *************
318 PRINTLEFT$(CD$,25)TAB(3)B$" - PRESS
ANY KEY";:GOSUB33
319 CLOSE2:CLOSE15:RETURN
320 REM *********** WRITE *************
321 FORK=0TO7:IFSP(K)=0THEN325
322 GOSUB339:GOSUB33:IFA<>13ANDA<>89THEN
SP(K)=0:GOTO324
323 SP(K)=2:X=1
324 B$="                ":GOSUB343
325 NEXTK:IFX=0THEN276
326 OPEN15,8,15:OPEN2,8,2,FI$+",S,W":X=0
327 INPUT#15,A,B$:IFA=63THENCLOSE2:OPEN2
,8,2,FI$+",S,A":X=1
328 INPUT#15,A,B$:IFATHENGOSUB318:GOTO27
6
329 FORK=0TO7:S=K:IFSP(K)=0THEN336
330 GETA$:IFA$=" "THENK=7:GOTO336
331 SP(K)=1:Y=1:GOSUB339:Y=0
332 B$=" << WRITING >>> ":GOSUB343
333 IFXTHENB$="   APPEND ! ":GOSUB343
334 FORJ=ADTOAD+62:B=PEEK(J):PRINT#2,B:N
EXTJ
335 B$="                ":GOSUB343
336 SP(S)=0:NEXTK
337 CLOSE2:CLOSE15:GOTO276
338 REM ********** ASK OK ? ***********
339 FORI=0TO7:S=I:AD=32768+64*S:IFSP(I)=
1THENI=7
340 NEXTI:POKEV+2*S,56:POKEV+2*S+1,150:P
OKEV+21,2^S
341 PRINTLEFT$(CD$,12)"      "S"     "
:IFYTHENRETURN
342 B$="   O.K. ?          "
343 PRINTLEFT$(CD$,16)"            "B$:
RETURN
344 REM *********** INTRO ************
345 POKE56576,5:POKE648,132:POKE56,124:R
EM SCREEN NACH 33792 - CHARACTERROM!!
```

```
346 PRINT"▒▒▒▒▒▒▒         ▣..............
.........."
347 PRINT"                              .
    ."
348 PRINT"   .    S P R I T E A I D +
349 PRINT"                              .
    ."
350 PRINT"                              .
351 PRINT"   .   WRITTEN  BY ▣ M A T A
N ▒  ."
352 PRINT"   .       MARBURG (1984)
353 PRINT"                              .
    ."
354 PRINT"   .   ▣ NO COPYRIGHTS !!
355 PRINT"                              .
    ."
356 PRINT"   ............................
......▣"
357 FORI=32768TO33344:POKEI,0:NEXT
358 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒";:FORI=1TO
19:PRINT" ";:FORII=1TO100:NEXTII,I
359 PRINT"▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒PRESS AN
Y KEY.▣"
360 GOTO33
361 REM ******** DATA LINES ************
362 POKEV+21,0:PRINT"▣▒.DATA GENERATOR.▣
"
363 PRINT"▒BEGIN WITH LINE (*10000) ? ";
:GOSUB30
364 IFA<49ORA>54THEN16
365 PRINT"▣"LEFT$(CD$,24)"▒  GENERATE DA
TA LINES   W A I T !"
366 ZN=(A-48)*10000:T=0:POKE646,PEEK(V+3
3):FORI=1TO2000:NEXT
367 IFT=8THEN10
368 AD=32768+64*T:PRINT"▣"ZN"REM SPRITE"
T
369 FORII=0TO6:ZN=ZN+1:PRINTZN"▒DATA";:F
ORI=0TO8
370 PRINTMID$(STR$(PEEK(AD+I+II*9)),2,3)
",";:NEXTI
371 PRINTCHR$(20):NEXTII
372 PRINT"ZN="ZN+1":T="T+1":GOTO367";:GO
TO447
373 REM******* SEND DISK COMMAND *******
374 GOSUB24:PRINT"▣▒ SEND DISK COMMAND.▣
":I=0
375 I=I+1:PRINTLEFT$(CD$,2*I+1)"=>";:C$=
""
376 GOSUB33:IFA=20ANDLEN(C$)>0THENPRINTA
$;:C$=LEFT$(C$,LEN(C$)-1):GOTO376
377 IFA=13THEN382
378 IFA=133THEN16
379 IFA<32ORA>127THEN376
380 C$=C$+A$:PRINTA$;
381 GOTO376
382 OPEN1,8,15,C$:INPUT#1,A,B$,C$:IFA=1T
HENPRINTLEFT$(CD$,25)C$;
383 GOSUB318:CLOSE1:IFA=133THEN16
384 PRINTLEFT$(CD$,25)"
                         ";:IFI<8THEN375
385 GOTO374
386 REM ******* COMMAND LIST ***********
387 GOSUB24:GOSUB156
388 PRINT"▒▣▒COMMAND LIST▣"LEFT$(CD$,3)
389 PRINT"▒SPRITES ▒        "
390 PRINT"▒ON/OFF  ▒=0-7 "
391 PRINT"INCR.    = I  "
392 PRINT"MULTI    = M  "
393 PRINT"COLOR    = C  "
394 PRINT"REPRO    = R  "
395 PRINT"▣ WORK ▒  = F1 "
396 PRINT"SLIP DWN= F7 "
397 PRINT"SLIP UP = F8 "
398 PRINT"BACKCOL.= F3 "
399 PRINT"HANDLE  = H  "
400 PRINT"SCREEN  = S  "
401 PRINT"PRINTER = P  "
402 PRINT"DATALINE= D  "
403 PRINT"▒FLOPPY  = F "
404 PRINT"SHOW $   = $  "
405 PRINT"DISK-CMD= @  ▣"
407 PRINT"KILL     = K  "
408 PRINT"QUIT     = Q  "
409 GOSUB33:GOSUB156:GOSUB172:GOTO68
410 REM ********* HANDLE ***************
411 POKEV+21,0:PRINT"▣▒.HANDLE SPRITES.▣
":Z=1:FORI=0TO7:S(I)=0:NEXT
412 PRINT"▒0-3   = SELECT SPRITE"
413 PRINT"'F7'  = FAST MOVE."
414 PRINT"▒USE 'CRSR' TO MOVE !"
415 GOSUB33:PRINT"▣▒       :  X=
Y=":GOTO418
416 PRINT"▣"TAB(10)PEEK(X)+255*SGN(PEEK(
V+16)AND2^S)"▒  "TAB(22)PEEK(Y)"▒  "
417 GOSUB33
418 IFA<48ORA>55THEN424
419 S=VAL(A$):PRINT"▒NO."S
420 X=V+2*S:Y=V+2*S+1
421 IFS(S)THEN423
422 POKEV+21,PEEK(V+21)OR2^S:S(S)=1:GOTO
416
423 POKEV+21,PEEK(V+21)AND255-2^S:S(S)=0
:GOTO416
424 IFA=134THENGOSUB103:GOTO416
425 IFA=17THENPOKEY,PEEK(Y)+ZAND255:GOTO
416
426 IFA=145THENPOKEY,PEEK(Y)-ZAND255:GOT
O416
427 IFA=29THEN434
428 IFA=157THEN436
429 IFA<>136THEN432
430 IFZ=1THENZ=5:GOTO416
431 Z=1:GOTO416
432 IFA<>133THEN416
433 POKEV+21,0:POKEV+16,0:GOTO16
434 IFPEEK(X)>=256-ZTHENPOKEV+16,PEEK(V+
16)OR2^S:POKEX,0
435 POKEX,PEEK(X)+ZAND255:GOTO416
436 IFPEEK(X)<=0+Z-1THENPOKEV+16,PEEK(V+
16)AND255-2^S:POKEX,255
437 POKEX,PEEK(X)-ZAND255:GOTO416
438 REM *********** KILL **************
439 PRINT"▣▒.CLEAR PROGRAM.▣"
440 PRINT"▒ARE YOU SURE ? ";:GOSUB30
441 IFA<>89ANDA<>13THEN16
442 PRINT"▣"LEFT$(CD$,24)"▒      CLEAR P
ROGRAM EXCEPT DATA !!"
443 POKE646,PEEK(V+33):T=0
444 PRINT"▒";:FORI=TTOT+7:PRINTI:NEXT
445 IFT>439THENPRINT"POKE 646,14:END":GO
TO447
446 PRINT"T="I":GOTO444"
447 POKE631,19:FORI=1TO9:POKE631+I,13:NE
XT:POKE198,10:END
READY.
```

Listing »Spriteaid+« (Schluß)

Screen Change

Ärgern Sie sich beim Erstellen umfangreicher Programme auch manchmal darüber, daß Sie nicht mehr genau wissen, wie ein bestimmter Programmteil aussieht? Dann werden Sie neu listen und der zuletzt bearbeitete Programmteil ist verschwunden. Die entsprechende Stelle muß wieder gesucht werden. Bis dahin hat man die zuerst gesuchte Stelle vielleicht schon wieder vergessen.

Die Routine »Screen Change« erweitert den Basicbefehlssatz dahingehend, daß der aktuelle Bildschirminhalt im RAM zwischengespeichert und bei Bedarf wieder zurückgeholt werden kann. So können bis zu vier Seiten abgelegt werden. Man hat also die Möglichkeit Notizen, oder andere Informationen, wie beispielsweise Listings oder Low-Resolution-Grafik, zu konservieren. Gleichermaßen sind Sie in der Lage, im Programmablauf eine Text- oder Grafikseite darzustellen, während eine weitere unsichtbar aufgebaut wird.

Listing Basic-Lader von »Screen Change«

```
10 S=0:I=0
20 FORI=49152TO49443
30 READA:S=S+A
40 REM     POKEI,A
50 NEXTI
60 IFS<>35160THENPRINT"DATEN SIND FALSCH
 EINGEGEBEN!"
70 SYS49152
71 DATA169,21,162,192,141,143,2,142,144,
2,169,181,162,192,141,8,3,142,9
72 DATA3,96,162,3,228,203,240,8,232,224,
7,208,247,76,72,235,228,197,240
73 DATA249,134,197,173,141,2,201,4,240,7
,201,6,240,6,76,72,235,32,174,192
74 DATA32,64,192,76,66,235,120,189,22,19
3,72,133,254,169,4,133,252,169,0
75 DATA133,251,133,253,72,198,1,162,4,16
0,0,177,253,112,6,72,177,251,145
76 DATA253,104,145,251,136,208,241,230,2
52,230,254,202,208,234,230,1,184
77 DATA104,170,172,35,193,192,32,240,4,1
04,76,152,192,189,29,193,41,191
78 DATA73,128,153,0,4,200,232,224,7,208,
240,104,74,74,41,3,9,176,153,0,4
79 DATA169,216,133,254,162,4,160,0,169,1
4,145,253,136,208,251,230,254,202
80 DATA208,246,88,96,8,104,9,64,72,40,96
,32,115,0,8,201,83,208,8,160,1,177
81 DATA122,201,69,240,7,40,32,121,0,76,2
31,167,32,115,0,162,3,32,115,0,221
82 DATA29,193,208,40,232,224,6,208,243,3
2,115,0,201,48,48,28,201,52,16,24
83 DATA41,15,24,105,3,170,32,115,0,201,4
4,208,11,32,115,0,201,72,240,7,201
84 DATA84,240,6,76,8,175,32,174,192,238,
35,193,32,64,192,206,35,193,32,115
85 DATA0,76,174,167,0,0,0,160,164,168,17
2,62,83,69,73,84,69,32
READY.
```

Nachdem der Basic-Lader gestartet wurde, ist das Programm automatisch initialisiert (SYS 49152). Durch gleichzeitiges Drücken der Tasten »CTRL« und »COMMODORE C« und einer f-Taste werden nun Video-RAM und entsprechender Seitenspeicher vertauscht. Normalerweise wird anschließend lediglich »Müll« auf dem Bildschirm zu sehen sein, da die betreffende Seite noch nicht beschrieben war. Drücken Sie nun aber die gleiche Kombination, oder einfach nur die Tasten »CTRL- und f-Taste« dann erscheint sofort der ursprüngliche Inhalt. Haben Sie die erste Kombination eingegeben wurden die Speicher wieder vertauscht, das heißt der Müll wurde zurückgeschrieben. Hingegen bleibt bei der zweiten Kombination der Seitenspeicher mit der gerade eingelesenen Information gefüllt. Dies läßt sich leicht überprüfen, indem der Bildschirm gelöscht wird, um daraufhin die zweite Kombination noch einmal einzugeben. Es zeigt sich sofort, daß der Inhalt unverändert geblieben ist. Dies ist solange der Fall bis die betreffende Seite mit Tastenkombination 1 »CTRL + C« eine andere Bildschirmseite aufnimmt. Das oben gesagte gilt für alle vier Funktionstasten.

Natürlich versagt diese Methode, wenn die gewünschte Funktion vom Basicprogramm her aufgerufen werden soll. Dazu wird der neue Befehl »SEITEx,m« eingeführt. Der Index x gibt dabei die Seitennummer und m den Modus an. Die Seitennummer darf dabei zwischen 0 und 3 liegen.m kann T oder H sein.

T bedeutet: Seiten tauschen; entsprechend Tastenkombination 1

H bedeutet: Seite holen; entsprechend Tastenkombination 2

Damit hat man die Möglichkeit Zwischenergebnisse abzulegen, eine Bedienungsanleitung einzublenden, um dann wieder in die normale Textseite zurückzukehren, Bildschirmmasken einzublenden und vieles mehr. Der interessierte Anwender wird sicher eine Fülle weiterer Einsatzmöglichkeiten entdecken. Doch nun einige Worte zum Programm selbst. Natürlich ist es auch möglich die Seiten über das Befehlswort im Basicdirektmodus aufzurufen. Dies hat jedoch den Nachteil, daß der Befehlstext sowie die READY-Meldung mit übertragen werden. Um dies zu umgehen wird die Tastaturabfrage über eine neue Routine geleitet, die die f-Tasten decodiert.

Der Befehl »SEITE« wird erkannt indem die Interpreterschleife meine Routine durchläuft. Dies kann im einzelnen dem Assemblerlisting entnommen werden.

Interessant ist in diesem Zusammenhang, daß die zusätzlich benötigten 4 KByte Video-RAM den freien Basicspeicherbereich nicht einschränken, da sie im normalerweise nicht nutzbaren Bereich hinter dem Interpreter liegen.

Zusätzlich habe ich eine Routine entwickelt, die über eine Befehlserweiterung das Laden und Speichern der Seiten auf Diskette zuläßt.

Über die normale Anwendung hinaus mögen folgende Anmerkungen von Nutzen sein:

Im CTRL-Modus wird in die rechte obere Bildecke die gewählte Seitennummer eingeblendet. Sollte dies als störend empfunden werden, so kann durch Ändern der letzten Zahl der letzten DATA-Zeile in eine beliebige andere die Einblendung ausgeschaltet werden.

Das Zurückschreiben der »32« führt wieder zur Standardausgabe. Über die Adresse 49313 (POKE 49313,F) läßt sich die Zeichenfarbe F der einzublendenden Seiten ändern. Gekoppelt mit dem Befehl SEITE ließen sich die verschiedenen Seiten in jeweils typischen Farben darstellen. Soll von vornherein eine bestimmte Zeichenfarbe eingestellt werden, so ist die »14« in der DATA-Zeile 79 dahingehend zu ändern.

Durch Anpassung der Zieladressen-Tabelle lassen sich die Seiten in jeden möglichen Speicherbereich legen. Dabei muß lediglich beachtet werden, ob eventuell das Basic-RAM oder ein darin liegendes Programm geschützt werden muß! Auf diese Weise lassen sich natürlich auch weit mehr Seiten anlegen.

Soll nun im Rahmen eines Basicprogramms eine Seite beschrieben werden, so kann dies wie gewohnt über POKEs erfolgen. Zu berücksichtigen sind lediglich die Startadressen der vier Seitenspeicher. Will man normalerweise den Bildschirm über POKE ansprechen, dann geht man von Adresse 1024 aus. Diese ersetzt man nun gegen die unten angegebenen und die gewählte Seite läßt sich unsichtbar beschreiben um bei Bedarf eingeblendet zu werden. Das Farb-RAM muß hier nicht neu beschrieben werden. Im folgenden nun die benötigten Adressen:

Seite 0 : 40960 bis 41959
Seite 1 : 41984 bis 42983
Seite 2 : 43008 bis 44007
Seite 3 : 44032 bis 45031

(Harald Soyka/rg)

Assembler Listing von »Screen Change«

```
90 OPEN4,4
100 SYS9*4096
110 *= $C000
120 .OPT P,OO
130 STO1      = $FB
140 STO2      = $FD
150 LSTX      = $C5
160 SFDX      = $CB
170 R6510     = $01
180 SHFLAG    = $028D
190 RETURN1   = $EB48
200 RETURN2   = $EB42
210 RETURN3   = $A7E7
220 VECTOR1   = $028F
230 VECTOR2   = $0308
240 CHRGET    = $0073
250 CHRGOT    = $0079
260 SYNTAX    = $AF08
270 ;
280 ; ***********    HARALD SOYKA    *****
*****
290 ; *********** HATTINGERSTR.685 *****
*****
300 ; ***********    463 BOCHUM 5    *****
*****
305 ; *********** TEL. 0234/411913 *****
*****
310 ;
320 ; DIESES PROGRAMM TAUSCHT DEN BILDSC
HIRMSPEICHER
330 ; MIT DEM BEREICH $A000-$B000. SOMIT
 KOENNEN BILD-
340 ; SCHIRMINHALTE ZWISCHENGESPEICHERT
WERDEN.
350 ; UEBER DIE FUNKTIONSTASTEN+(CTRL) K
ANN LAUFEND
360 ; ZWISCHEN VIER BILDSCHIRMSEITEN GEW
ECHSELT WERDEN.
370 ;
380 ;
390 ; ********** PRG INITIALISIEREN ****
*****
400 LDA #<START
410 LDX #>START          ;TASTATURABFRAGE
420 STA VECTOR1          ;UEBER EIGENE
430 STX VECTOR1+1        ;ROUTINE LEITEN
440 ;
450 LDA #<INTER
460 LDX #>INTER          ;INTERPRETER
470 STA VECTOR2          ;UEBER EIGENE
480 STX VECTOR2+1        ;ROUTINE LEITEN
490 RTS
500 ;
510 ; *** AUF GEDRUECKTE F-TASTEN PRUEFE
N ***
520 START LDX #$03
530 S0 CPX SFDX
540 BEQ FOUND
550 INX
560 CPX #$07
570 BNE S0
580 BACK  JMP RETURN1
590 ;
600 FOUND CPX LSTX
610 BEQ BACK
620 STX LSTX
630 LDA SHFLAG               ;CONTROL TASTE
640 CMP #$04                 ;GEDRUECKT
650 BEQ M1
660 CMP #$06
670 BEQ M2
680 JMP RETURN1
690 M1 JSR VFL               ;V-FLAG SETZEN
700 M2 JSR CHAN              ;CHANGE SRCEEN
710 JMP RETURN2
720 ;
730 ; *** ZEIGER FUER VERSCHIEBEROUTINE
SETZEN ***
740 ;
750 CHAN SEI
760 ;
770 LDA SEITE,X              ;NUMMER DER TASTE
780 PHA
790 STA STO2+1               ;IN X
800 LDA #04
810 STA STO1+1
820 LDA #$00
830 STA STO1
840 STA STO2
850 PHA
860 ;
870 ;*********** RAM  AUSTAUSCHEN ******
*****
880 RAM DEC R6510            ;$A000 RAM FRE
IGEBEN
890 LDX #4                   ;4 BLOECKE
900 LDY #0
910 S2 LDA (STO2),Y          ;VON  $AXXX
920 BVS M3
930 PHA
940 LDA (STO1),Y             ;VON  $0400
950 STA (STO2),Y             ;NACH $AXXX
960 PLA
970 M3 STA (STO1),Y          ;NACH $0400
980 DEY
990 BNE S2
1000 ;
1010 BLOCK INC STO1+1        ;ZEIGER
1020 INC STO2+1              ;HOCHZAEHLEN
1030 DEX
1040 BNE S2
1050 INC $01
1060 CLV
1070 ;
1080 ; ******** SEITENNUMMER EINTRAGEN *
******
1090 PLA
1100 TAX
1110 LDY TEXT+6              ;AUF MODUS
1120 CPY #$20                ;PRUEFEN
1130 BEQ S1
1140 PLA
1150 JMP FARB                ;FARBE SETZEN
1160 S1  LDA TEXT,X
1170 AND #%10111111
1180 EOR #$80                ;IN BILDSCHIRM
1190 STA $0400,Y             ;SCHREIBEN
1200 INY
```

```
1210 INX
1220 CPX #$07                ;ALLE ZEICHEN
1230 BNE S1
1240 PLA
1250 LSR : LSR
1260 AND #%00000011          ;SEITENNUMMER
1270 ORA #%10110000
1280 STA $0400,Y             ;EINFUEGEN
1290 ;
1300 ; ********* FARB-RAM AUFUELLEN ***
******
1310 FARB LDA ##$D8               ;ZEIGER AUF
1320 STA ST02+1              ;FARB-RAM
1330 LDX #4
1340 LDY #0
1350 LDA #14                 ;FARBE NR. 14
1360 S3  STA (ST02),Y
1370 DEY
1380 BNE S3                  ;IN RAM
1390 INC ST02+1
1400 DEX
1410 BNE S3                  ;SCHREIBEN
1420 CLI
1430 RTS
1440 ;
1450 ; *********** V-FLAG SETZEN *******
******
1460 VFL PHP
1470 PLA
1480 ORA #%01000000
1490 PHA
1500 PLP
1510 RTS
1515 ;
1520 ; ************ BASIC-CODE SUCHEN **
******
1530 INTER JSR CHRGET             ;ZEICHEN HO
LEN
1540 PHP
1550 CMP ##$53
1560 BNE PL
1570 LDY ##$01
1580 LDA ($7A),Y             ;ZEICHEN HOLEN
1590 CMP ##$45
1600 BEQ TAUSCH
1610 PL  PLP
1620 JSR CHRGOT
1630 JMP RETURN3                  ;BEFEHL AUSFUEHRE
N
1635 ;
1640 TAUSCH JSR CHRGET
1650 LDX ##$03                    ;BEFEHLSCODE
1660 S4 JSR CHRGET
1670 CMP TEXT,X                   ;PRUEFEN.
1680 BNE SY                       ;EINGABE FEHLER
1690 INX
1700 CPX ##$06
1710 BNE S4
1720 JSR CHRGET                   ;SEITENNR.
1730 CMP ##$30
1740 BMI SY
1750 CMP ##$34                    ;HOLEN
1760 BPL SY
1770 AND ##$0F                    ;UND AUFBEREITEN
1780 CLC
1790 ADC ##$03
1800 TAX
1810 JSR CHRGET                   ;NAECHSTES ZEICHE
N
1820 CMP ##$2C
1830 BNE SY
1840 JSR CHRGET
1850 CMP ##$48
1860 BEQ I1
1870 CMP ##$54
1880 BEQ I2
1890 SY   JMP SYNTAX              ;SYNTAX ERROR
1900 I1   JSR VFL                 ;V-FLAG SETZE
N
1910 I2   INC TEXT+6              ;EINBLENDUNG
1920 JSR CHAN
1930 DEC TEXT+6                   ;BLOCKIEREN
1940 JSR CHRGET
1950 JMP $A7AE
1955 ;
1960 ; **** HIGH BYTES DER ZIELADRESSEN
 *****
1970 SEITE .BYTE $00,$00,$00,$A0,$A4,$A8
,$AC
1980 ;
1990 ; ************* BILDEINTRAG *******
******
2000 TEXT .ASC ">SEITE "
READY.
```

Assembler Listing von »Screen Change« (Schluß)

TIPS & TRICKS

Verstimmter C 64?

Benutzt man für ein Musikstück die im Commodore 64-Handbuch angegebenen High- und Low-Bytes, um die Töne zu POKEn, dann klingen sie häufig unrein oder »verstimmt«. Das vermeidet man, indem man die Low-Bytes neu festlegt. Man kann sie nach der folgenden Formel berechnen:

Low-Byte = Frequenz * 17 - High-Byte * 256

Ist das Ergebnis negativ, dann nimmt man ersatzweise diese Formel:

Low-Byte = Frequenz * 17 - (High-Byte - 1) * 256

(Roger Limberg)

VC 20 — Tips

Umschalten des VC 20 auf die Grundversion bei eingesteckter Speichererweiterung:

POKE 642,16 : POKE 644,30 : POKE 648,30 : SYS 64824

Mit POKE 55,30 : SAVE »(Name)« kann ein SAVE-Schutz umgangen werden, mit dem viele Programme geschützt sind.

Der Befehl POKE 36867,48 erzeugt eine zusätzliche Zeile unterhalb des normalen Bildschirms, die während des gesamten Programms stehen bleibt und nur über POKE-Befehle zugänglich ist.

(Frank Pachollek)

»Fort Apocalypse« bezwungen

Fort Apocalypse, eines der interessantesten und schwierigsten Spiele für den Commodore 64, wurde bislang nur von ausgefuchsten Spezialisten bezwungen. Der Grund dafür ist, daß zur Bewältigung der vielen Hürden und Aufgaben nur sechs Hubschrauber zur Verfügung stehen.

Dies kann man aber grundlegend ändern, indem man nach dem Laden den folgenden Befehl eingibt:
POKE 36339,153

Jetzt das Programm starten, und schon hat man 98 (!) Hubschrauber zur Verfügung. Da kann doch kaum noch was schiefgehen ...

(Klaus Kierblewski)

Basic-Programme retten

Ein versehentlich mit »NEW« oder durch einen RESET gelöschtes Programm kann beim VC 20 durch Eingabe der folgenden Befehle im Direktmodus wieder zurückgeholt werden:

POKE 46, PEEK(56) - 1 : POKE 45, PEEK(55) + 247 : CLR »Return«

POKE PEEK(44) * 256 + PEEK(43) + 1, PEEK(44) »Return« 63999 »Return«

FOR I = PEEK(44) * 256 + PEEK(43) TO PEEK(46) * 256 + PEEK(45) : IF PEEK(I) OR PEEK (I + 1) OR PEEK(I + 2) THEN NEXT »Return«

POKE 45, (I + 3) AND 255 : POKE 46, (I + 3) / 256 : CLR »Return«

Unter Umständen erhält man jetzt eine Fehlermeldung, aber das Programm ist jedenfalls wieder da.

(Ralf Berle)



Fehlersuche leicht gemacht

Fehlersuche in Programmen ist nicht einfach. Der LIST-Befehl des C 64 bietet dabei auch nicht gerade eine große Erleichterung. Dieses Programm schafft Abhilfe.

Wen ärgert es nicht, wenn man nach mühseliger Abtipperei eines Programmlistings feststellen muß, daß es nach »RUN« nicht richtig läuft? Nun geht es an die Fehlersuche und die ist mit dem normalen LIST-Befehl nicht gerade einfach. Wenn man keine gute Reaktion an den Tag legt, sind einige Zeilen oben aus dem Bildschirm verschwunden, bevor man »RUN/STOP« gedrückt hat. Und hat man erst einmal RUN/STOP gedrückt, dann muß man wieder einen neuen LIST-Befehl mit Zeilennummer eingeben, um weiter aufzulisten. Auch die »CTRL«-Taste bringt keinen besonderen Vorteil, denn so oft ich mich auch bemüht habe, das Rennen gegen den C 64 zu gewinnen, bis heute ist es mir noch nicht gelungen, ein gedrucktes Listing mit dem des C 64 so zu vergleichen, daß ich ohne RUN/STOP ausgekommen wäre. Dies gilt im verstärkten Maße, wenn DATA-Zeilen mit im Spiel sind.

Um nun dieses Problem zu lösen, habe ich das LIST-STOP-Programm entwickelt. Mit diesem ist es möglich, ein Listing zu kontrollieren oder zu vergleichen, ohne in Zeitnot zu geraten oder jedesmal erneut einen LIST-Befehl einzugeben.

Ist das Programm aktiviert, dann wird die Routine mit »Pfeil links« und dann RETURN — entspricht CHR$ (95) und CHR$ (13) — aufgerufen. Es wird jetzt solange aufgelistet, bis ein Bildschirm voll ist. Hat der C 64 gestoppt, so ist er wieder im Eingabemodus und man kann alle Editiermöglichkeiten in Anspruch nehmen, um Fehler zu korrigieren, Zeilen einzufügen oder zu entfernen.

Will man nun das Listen weiterlaufen lassen, braucht man nur CHR$ (133) beziehungsweise F1 zu drücken und es geht bei derZeilennummer weiter, bei der vorher gestoppt wurde (auch wenn etwas verbessert wurde!), so lange, bis wieder ein Bildschirm voll ist.

Zu beachten ist, daß der neue LIST-STOP-Befehl einen LIST-Befehl simuliert und man ihn auch als solchen behandeln kann. Um bestimmte Zeilennummern aufzulisten, kann folgende Eingabe verwendet werden:
»Pfeil links« 20 - 130, RETURN.

Jetzt werden nur die Zeilen zwischen 20 und 130 mit der LIST-STOP-Routine behandelt.

Der normale LIST-Befehl kann weiterhin benutzt werden. Die zwei Zeichen, die mehrfach zu sehen sind (CHR$ (113) und CHR$ (64), ≙ geschlossener Kreis und »Klammeraffe«), sind Steuerzeichen für das Programm (keine Fehler).

Der Grund dafür, daß CHR$ (95) »Pfeil links« als Basic-Befehl erkannt wird, liegt darin, daß mit dem Aufruf des Programms durch SYS 40704 der Vektor für die Basic-Befehlsadresse auf meine eigene Abfrageroutine gesetzt wird. Erkennt diese Routine das Zeichen, wird der LIST-Befehl ausgeführt.

Programmierhinweis

Nachdem das Programm abgetippt wurde, sollte man es unbedingt vor dem ersten Lauf abspeichern, da sich das Basicprogramm selbsttätig löscht, nachdem LIST-STOP aktiviert wurde.

Ist ein Fehler in den DATA-Zeilen so meldet der Computer den Fehler und bricht das Programm ab, damit man den Fehler beheben kann. Als Erleichterung habe ich für jede DATA-Zeile die Summe aufgeführt. Das Maschinenprogramm ist ab dez. 40704 abgelegt und wird durch Zeile 10 vor dem Überschreiben geschützt. (Manfred Selke/rg)

Die Prüfnummern für die einzelnen DATA-Zeilen

Prüfsummen

Zeile	Summe	Zeile	Summe
111	1707	117	2745
112	1935	118	1938
113	2432	119	2446
114	2996	120	1863
115	2252	121	2143
116	2134	122	226

```
0 REM ************************
1 REM ***    LIST - STOP   ***
2 REM ***       C-64       ***
3 REM ***     (C) 1984     ***
4 REM *** BY MANFRED SELKE ***
5 REM ***   NEUMUENSTER    ***
6 REM ************************
7 REM
8 REM
10 POKE56,159:POKE55,0:CLR:A=40703
20 A=A+1:READB:S=S+B:IFB=-1THEN40
30 POKEA,B:GOTO20
40 IFS<>24817THENPRINT"FEHLER IN DATA'S
!":END
50 SYS40704:NEW
100 REM
105 REM *** DATA'S FUER LIST-STOP ***
110 REM
111 DATA169,19,141,8,3,169,159,141,9,3,1
41,1,3,169,185,141,0,3,96,32,115
112 DATA0,201,95,208,33,169,155,141,0,2,
169,105,141,20,3,169,159,141,21,3
113 DATA169,81,141,254,159,141,112,7,169
,24,133,214,32,121,0,76,231,167,201
114 DATA64,208,246,173,254,159,201,81,20
8,239,173,252,159,133,20,173,253
115 DATA159,133,21,32,19,166,173,250,159
,133,20,173,251,159,133,21,169,81
116 DATA141,112,7,169,24,133,214,76,201,
166,173,0,4,201,81,208,39,165,20
117 DATA141,250,159,165,21,141,251,159,1
60,2,177,95,141,252,159,200,177,95
118 DATA141,253,159,169,0,133,20,133,21,
169,32,141,0,4,169,154,141,20,3,76
119 DATA49,234,165,197,201,4,208,247,169
,105,141,20,3,32,68,229,169,64,141
120 DATA119,2,169,13,141,120,2,169,2,133
,198,76,49,234,24,165,20,5,21,201
121 DATA0,240,15,169,49,141,20,3,169,234
,141,21,3,169,0,141,254,159,76,139
122 DATA227,0,-1
READY.
```

Der Basic-Lader von »LIST-STOP«

Haben Sie den Bogen raus?

Wenn Sie mit Hilfe von Simons Basic Halb- und Viertelkreise, sowie Rauten, Dreiecke, Fünfecke und so weiter zeichnen wollen, müssen Sie den ARC-Befehl völlig durchschaut haben. Um aber den ARC-Befehl wirklich zu verstehen, sollte man mit CIRCLE beginnen.

Wie Sie wissen, gehen wir bei HIRES von einem Grid aus, das aus 64000 Einzelpunkten besteht. Da das »erste« beim C 64 immer Null heißt, lauten die Adressen:
waagerecht (x-Achse): 0 - 319
senkrecht (y-Achse): 0 - 199

Bei HIRES/MULTI hingegen ist jeder der 32000 Punkte doppelt so breit, so daß die Adressen (x-Achse) dann 0 - 159 lauten.

Als erstes brauchen wir die Koordinaten x, y, welche den Mittelpunkt des Kreises bestimmen. Laut Handbuch von Simons Basic stellt der »Kreis eine Sonderform der Ellipse dar«, so daß der CIRCLE-Befehl auch für »ovale Kreise« benutzt werden kann. Daher müssen wir zwei Radien angeben (das ist eine gedachte Linie vom Kreismittelpunkt zum Kreisrand), und zwar einen waagerechten (x1) und einen senkrechten (y1). Somit sieht das Format nun so aus:

CIRCLE x,y,x1,y1,...

Aber, nach y1 fehlt noch eine Angabe. Wir haben noch nicht gesagt, wie die Figur gezeichnet werden soll! Es gibt drei Möglichkeiten:

zeichentyp:HIRES+MULTI	
0=IMMER HINTERGRUNDFARBE	
HIRES	MULTI
1:Zeichenfarbe	1:Zch.farbe NR.1
2:invertiert	2:Zch.farbe NR.2
NICHTS	3:Zch.farbe NR.3
NICHTS	4:invertiert: Farben 0 und 3 Farben 1 und 2

Bild 1. Tabelle/Zeichentyp:HIRES + MULTI

a) Hintergrundfarbe (zum Beispiel wenn wir in einen Block nachträglich hineinzeichnen möchten),
b) Zeichenfarbe (bei HIRES nur eine, bei MULTI können wir unter drei wählen), oder
c) Invertiert (das heißt: wo bereits ein Punkt gesetzt ist, zum Beispiel durch eine mit PAINT gefüllte Fläche, wird jetzt gelöscht; wo keiner ist, wird einer gesetzt).

Der Parameter, der die Entscheidung über den Zeichenmodus trägt, heißt: Zeichentyp. Die Tabelle (Bild 1) faßt alle möglichen Zeichentyp-Zahlen zusammen. Jetzt ist unser CIRCLE-Befehl vollständig:

CIRCLE x,y,x1,y1,ZT

Wenn Sie ein wenig herumprobieren, dann werden Sie feststellen: Um einen »richtigen« Kreis zu bekommen, muß bei HIRES x1 gleich y1 sein! Da bei MULTI jeder Punkt doppelt so breit ist, muß der x-Radius (x1) die Hälfte von y1 betragen.

Nun wollen wir mit dem Experimentieren anfangen. Als erstes ein START-Programm:

10 COLOUR 14,14: HIRES 0,14
(Jetzt wird mit schwarz auf hellblau gezeichnet.)
15 REC 0,0,319,199,1

Dieses RECHteck zeichnet einen Rahmen um den gesamten Bildschirm.

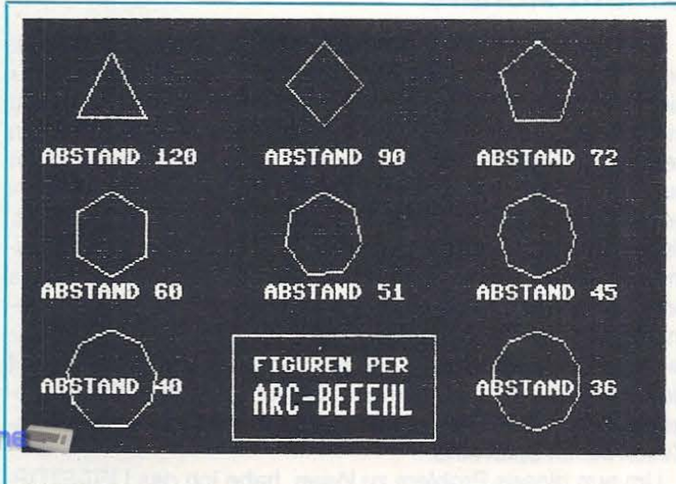


Bild 2. Verschiedene Figuren per ARC-Befehl

Aber noch können wir nichts sehen, weil das Bild zu schnell verschwindet. Testen Sie dies einmal durch RUN. Deshalb gebe ich von Anfang an immer eine Programmzeile mit höherer Nummer ein:

90 WAIT 653,2

Dies ist ein normaler C 64-Befehl; er bewirkt ein Warten, bis jemand die CBM (C=)-Taste drückt. Durch diesen Befehl können Sie Ihre derzeitigen Versuche immer am Bildschirm betrachten und dann am Programm weiterarbeiten! Wenn Sie jetzt noch einmal RUN eintippen, dann sehen Sie das Rahmen-REC! Jetzt können wir weitermachen:

20 CIRCLE 159,99,50,50,1

Bitte versuchen Sie, ehe Sie jetzt RUN eintippen, vor Ihrem geistigen Auge sich vorzustellen, was gleich zu sehen sein wird. Erstens: Wo ist der Kreis? Zweitens: Ist es ein »richtiger« Kreis oder eine Ellipse? Wenn Sie sich ein »Bild« machen können, dann erst durch RUN Ihren Eindruck überprüfen:

Aha! Es ist ein Kreis in der Mitte des Bildschirms. Da wir HIRES mit gleichen Längen bei x1 und y1 haben, ist es ein »echter« Kreis.

Jetzt probieren Sie bitte weiter:
a) Verlängern Sie x1, das heißt den waagerechten Radius! Resultat: Ein »Kreis«, der breiter wird, also eine waagerechte Ellipse!
b) Verlängern Sie y1, machen Sie also eine senkrecht stehende Ellipse daraus!

Jetzt geben Sie in Zeile 10 noch einen Doppelpunkt plus MULTI 0,6,0 ein.

Frage: Was wird mit unserem Rahmen-REC passieren, nachdem wir auf MULTI umgeschaltet haben? Bitte testen Sie dies durch RUN!

Da wir bei MULTI ja nur 160 doppelt so breite Punkte wie bei HIRES haben und da der letzte Punkt auf der x-Achse 159 heißt: Wie muß jetzt der REC-Befehl verändert werden, damit Sie wieder einen kompletten Rahmen erhalten?

Richtig: 319 muß in 159 umgewandelt werden.

Lassen Sie jetzt beide CIRCLE-Kombinationen von oben nochmal mit MULTI laufen: Die erste ergibt natürlich keinen »richtigen« Kreis mehr, denn 50 Punkte auf jeder Achse bedeuten ja 50 doppelt-breite Punkte bei x1, daher muß dies zu einer Ellipse führen!

Da die Erklärungen für ARC davon ausgehen, daß Sie den CIRCLE-Befehl voll und ganz verstanden haben, sollten Sie im Zweifelsfall noch ein wenig herumspielen, ehe Sie weiterlesen!

ARC heißt »Bogen« (vgl. ARC de Triomphe in Paris: Triumph-Bogen). Der ARC-Befehl zeichnet verschiedene Figuren, deren Außenlinie wir als Bogen verstehen müssen. Beginnen wir mit einem Kreis:

Als erstes brauchen wir x,y (wie bei CIRCLE). Als nächstes müssen wir zwei Angaben machen, die beim CIRCLE-Befehl bereits »eingebaut« sind, nämlich zwei Winkelangaben:

a) einen START-Winkel und
b) einen END-Winkel.

Wir beginnen unser Experiment mit den bei CIRCLE automatisch festgelegten Werten, nämlich

a) START-Winkel gleich 0
b) END-Winkel gleich 360

Eine solche Figur wird immer eine ganze Figur werden, also in unserem Beispiel zunächst ein ganzer Kreis. Veränderungen bei START- und END-Winkel werden auch Halb- oder Viertelfiguren liefern. Aber zunächst, weiter mit SW 0 und EW 360. Bis jetzt haben wir festgelegt:

ARC x,y,SW,EW,...

Die nun folgende Zahl ist immens wichtig: Sie bestimmt den ABSTAND der einzelnen Punkte auf der Bogenlinie. Beginnen

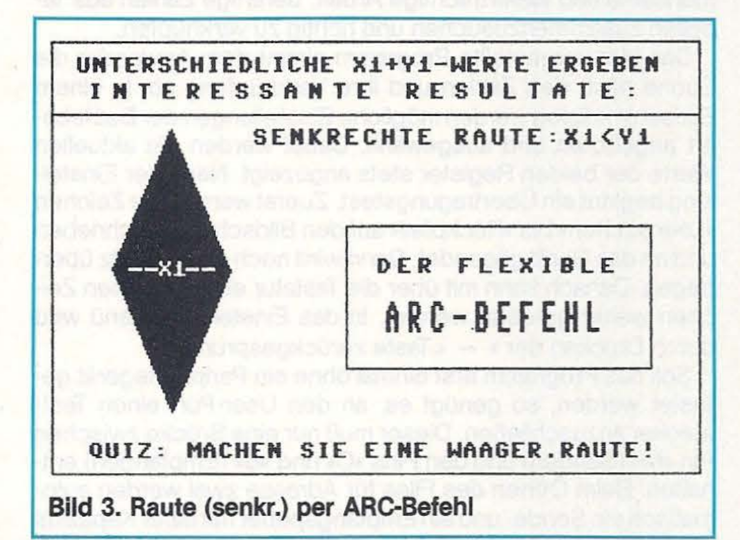


Bild 3. Raute (senkr.) per ARC-Befehl

wir mit 3, das heißt alle drei Punkte »weit« wird auf unserer Bogenlinie ein Punkt gesetzt:

ARC x,y,SW,EW,3,...

Nun fehlen nur noch:

a) Radius x-Achse (x1),
b) Radius y-Achse (y1) sowie
c) ZT = Zeichentyp.

Die drei Parameter kennen Sie von CIRCLE her ja sehr gut!

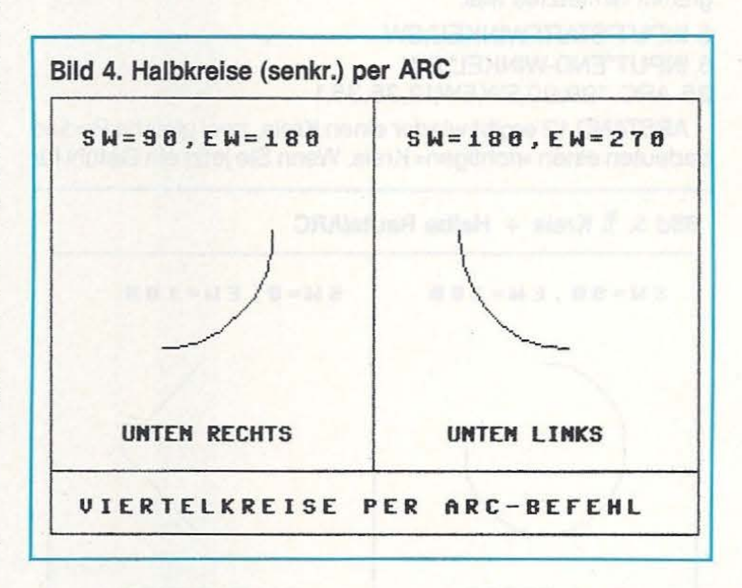
Somit können wir beginnen. Nehmen Sie das kleine START-Programm von vorhin. Legen Sie den MULTI-Befehl durch ein vorgeschaltetes REM vorläufig »auf Eis« und geben dann ein:

20 CIRCLE 90,99,35,35,1

25 ARC 190,99,0,360,3,35,35,1

Es folgt RUN und erhöhte Aufmerksamkeit: Denn, wiewohl beide Befehle einen Kreis zeichnen, gibt es einen wichtigen Unterschied, den Sie am Bildschirm beobachten können. Welchen?

Bild 4. Halbkreise (senkr.) per ARC



Nun, haben Sie es gemerkt? Der ARC-Kreis wurde viel langsamer gezeichnet. Warum? Weil wir einen ABSTAND von drei gewählt hatten. Beim CIRCLE-Befehl ist nämlich ein ABSTAND von zwölf bereits »eingebaut«, deshalb ist der CIRCLE-Kreis schneller erstellt. Denn wenn jeder dritte Punkt vom Computer berechnet und gezeichnet werden muß, dann sind dies viel mehr Punkte auf der Gesamtfigur, als wenn nur jeder zwölfte Punkt gefordert wird!

Jetzt kommt ein ganz wesentliches Experiment: Bitte ändern Sie Ihr Programm um: Erstens löschen Sie den CIRCLE-Befehl, zweitens addieren Sie:

5 INPUT "ABSTAND"; AB

In Zeile 25 ändern Sie den ABSTANDs-Parameter von 3 in AB um. Spielen Sie jetzt bitte mit folgenden ABSTANDs-Zahlen:

12 (wie beim CIRCLE-Befehl)
18 (noch Kreis-artig!)
36 (noch immer ein Kreis?)
40 (Resultat?)
60 (Resultat?)
72 (Resultat?)
90 (Resultat?)
120 (Wer hätte das gedacht?)

(Bild 2 zeigt die wichtigsten ABSTANDs-Werte, zum Nachschlagen.)

Sie sehen also: Der ARC-Befehl ist einer der interessantesten Befehle! Aber noch immer haben wir ihn kaum zu nutzen begonnen. Nehmen wir im folgenden einen ABSTAND von 90 (Raute) und spielen ein wenig weiter. Wenn Sie also die Zeile fünf und sechs wie folgt verändern:

5 INPUT"X-RADIUS";X1
6 INPUT"Y-RADIUS";Y1

In Zeile 25 tauschen Sie AB durch 90 aus und setzen an die Stelle von x1 den Variablen-Namen X1 und an die Stelle von y1 den Variablen-Namen Y1:

25 ARC 190,99,0,360,90,X1,Y1,1

Jetzt können Sie Rauten produzieren, die schmal und breit oder schmal und lang sind! (Bild 3)

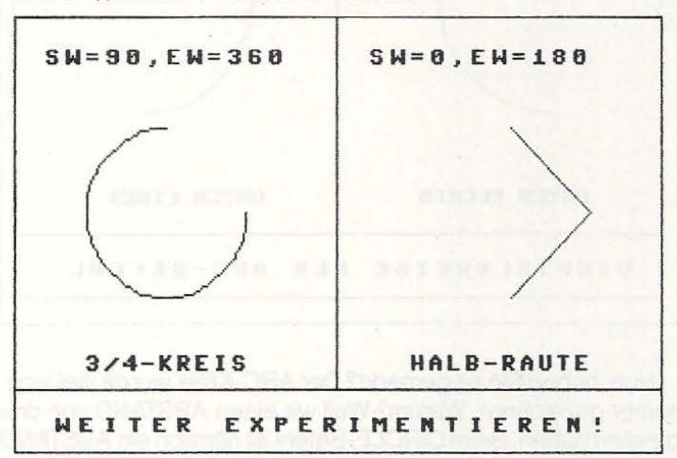
Aber auch das ist noch nicht alles! Geben Sie statt ABSTAND 90 mal 120 ein und probieren wieder per INPUT mit verschiedenen Radius-Zahlen!

Und zuletzt beginnen wir, die START- und END-Winkel-Zahlen zu verändern! Denn: START 0 und END 360 ergibt ja eine voller Figur. Was aber, wenn Sie einen Halbkreis oder eine Halb-Raute wollen? Obere oder untere Hälfte? Waagerecht oder senkrecht »zerschnitten«? Bitte ändern Sie unser Programm ein letztes Mal:

```
5 INPUT"START-WINKEL";SW
6 INPUT"END-WINKEL";EW
25 ARC 199,90,SW,EW,12,35,35,1
```

ABSTAND 12 ergibt wieder einen Kreis, zwei gleiche Radien bedeuten einen »richtigen« Kreis. Wenn Sie jetzt ein Gefühl für

Bild 5. ¾ Kreis + Halbe Raute/ARC



die Möglichkeiten, welche Ihnen die START- und END-Winkel-Zahlen bieten, gewonnen haben(siehe Bild 4), dann:

a) Verändern Sie den ABSTAND wieder (siehe Bild 5)
b) Verändern Sie den x-Radius (x1)
c) Verändern Sie den y-Radius (y1)

Dadurch, daß Sie immer nur einen Parameter verändern, bekommen Sie für jeden einzelnen dieser Parameter ein »Gefühl«; somit werden Sie den ARC-Befehl durchs Ausprobieren wirklich in den Griff bekommen!



Bild 6. Einladung zu einer Party

In Verbindung mit REC für den Rahmen und Text-Befehlen, kann mit ARC in Null-Komma-Nix sogar »ernstlich« gearbeitet werden: Visitenkarten, Briefbögen, Übersichten, Einladungen (Bild 6) und so weiter. Wenn Sie einen Drucker haben, der den COPY-Befehl des Simons Basic unterstützt, oder aber mit einem Maschinenprogramm per SYS-Befehl Ihre Bildschirme auf Papier überträgt, dann steht Ihren »Fotosatz«-Experimenten nichts mehr im Wege!

(Vera F. Birkenbihl/aa)

Die RS232-Schnittstelle am VC 20

Der VC 20 enthält im Betriebssystem bereits alle notwendigen Routinen zur Verwaltung einer RS232-Schnittstelle. Für den Anschluß von Peripheriegeräten wird auf den User-Port nur noch eine Erweiterungsplatine zur Anpassung der Signalpegel gesteckt.

Auf dem Markt erhältlich sind sowohl Schaltungen nach der V.24-Norm als auch nach der TTY-Norm (20 mA). Nach dem richtigen elektrischen Verbinden der Geräte ist die Einstellung der Parameter für die Zeichenübertragung im Computer dem angeschlossenen Gerät entsprechend vorzunehmen.

Eingestellt werden müssen die Baudrate, die Anzahl der Datenbits, die Anzahl der Stoppbits, die Art der Paritätsüberprüfung und das Rückmeldeverhalten. Im VC 20, wo diese Schnittstelle unter der Gerätenummer 2 verwaltet wird, müssen beim Öffnen des Übertragungskanals in zwei Registern, dem Kontrollregister und dem Befehlsregister, Zahlen zur Einstellung der Betriebsart übergeben werden. Es ist stets eine mühsame und fehlerträchtige Arbeit, derartige Zahlen aus Tabellen zusammenzusuchen und richtig zu verknüpfen.

Das hier vorgestellte Programm nimmt dem Anwender die Suche nach den Zahlen und ihre Verknüpfung ab. In einem Bildschirmdialog werden mögliche Einstellungen der Betriebsart angeboten und ausgewählt. Dabei werden die aktuellen Werte der beiden Register stets angezeigt. Nach der Einstellung beginnt ein Übertragungstest. Zuerst werden die Zeichen »Leerzeichen« bis »Rückpfeil« auf den Bildschirm geschrieben und an das Gerät gesendet. Dann wird noch ein Testsatz übertragen. Danach kann mit über die Tastatur eingegebenen Zeichen weiter getestet werden. In das Einstellungsmenü wird durch Drücken der » ← «-Taste zurückgesprungen.

Soll das Programm erst einmal ohne ein Peripheriegerät getestet werden, so genügt es, an den User-Port einen Teststecker anzuschließen. Dieser muß nur eine Brücke zwischen Pin »M« (Senden) und den Pins »C« und »B« (Empfangen) enthalten. Beim Öffnen des Files für Adresse zwei werden automatisch ein Sende- und ein Empfangspuffer mit einer Kapazität von je 256 Zeichen bereitgestellt. Über die Kurzschlußbrücke werden die gesendeten Daten in den Empfangspuffer übertragen. Dort können sie ausgelesen und weiter verarbeitet werden. In diesem Programm werden sie einfach auf den Bildschirm geschrieben.

So arbeitet des Testprogramm

In Zeile 140 werden Anfangswerte in die Variable K(I) für die veränderlichen Parameter der Schnittstelle eingegeben. Die Datazeilen 150 bis 240 enthalten Textblöcke für den Bildschirmdialog. Nach dem Einschalten der Repeat-Funktion für alle Tasten in Zeile 250 beginnt der Dialogabschnitt in Zeile

Testprogramm zum Initialisieren einer seriellen Schnittstelle

```
100 REM RS 232 SCHNITTSTELLE
110 REM 13.02.84
120 REM (C) H.-J. KELLERMANN
130 REM ANFANGSWERTE FESTLEGEN
140 K(0)=0:K(1)=6:K(2)=0:K(3)=0:K(4)=0:K
(5)=0:K(6)=0
150 DATA6,"","TEST-ROUTINE","BAUD-RATE",
"DATENWORT-LAENGE","STOP-BITS"
160 DATA"HANDSHAKE","UEBERTRAGUNGSART","
PARITAETSPRUEFUNG"
170 DATA10,"BAUD"," ANW.  ","  50   ","  7
5   ","  110   ","  134.5","  150   "
180 DATA"  300   ","  600   ","1200  ","1800
  ","2400  "
190 DATA3," BITS",8,7,6,5
200 DATA1," STOP-BITS",1,2
210 DATA1,-DRAHT,3,X
220 DATA1,DUPLEX,VOLL,HALB
230 DATA7,,OHNE PAR.PRUEF.,UNGER.PARIT.,
OHNE PAR.PRUEF
240 DATAGERAD.PARIT.,OHNE PAR.PRUEF,8.BI
T AUF 1,OHNE PAR.PRUEF,8.BIT AUF 0
250 POKE650,128:REM ALLE TASTEN REPEAT
260 REM
270 REM MENUE ZUR PARAMETERAENDERUNG
280 RESTORE:FORJ=0TO6:READK%(J)
290 READK$:FORI=0TOK%(J):READK$(I,J):K$(
I,J)=K$(I,J)+K$
300 NEXTI:NEXTJ
310 J=0:K(J)=K(J)+1:IFK(J)=7THENK(J)=0
320 GOSUB680:K$="REGISTERINHALTE█████████
----------------"
330 K$=K$+"██████████████████KR="+STR$(KR
)+"█  █BR="+STR$(BR)+"██"
340 GOSUB520:J=K(J):IFJ>0THENK$=K$(J,0):
GOSUB520:GOTO310
350 REM
360 REM TESTROUTINE ABARBEITEN
370 GOSUB710:GOSUB690:GOSUB660:CLOSE1:GO
SUB700
380 GOSUB680:OPEN1,2,0,CHR$(KR)+CHR$(BR)
:GOSUB700
390 GET#1,I$:REM PUFFER INITIALISIEREN
400 PRINT"█";:REM OPTISCHE KENNUNG FUER
SENDEN
410 FORI=32TO95:PRINTCHR$(I);:PRINT#1,CH
R$(I);:NEXT:REM ZEICHENSATZ
420 PRINT#1:PRINT:PRINT#1:PRINT
430 PRINT#1,"THE QUICK BROWN FOX JUMPS O
VER THE LAZY DOG 0123456789"
440 PRINT#1:POKE204,0:REM CURSOR BLINKEN
 EIN
450 GET#1,I$:IFI$=""THEN480
460 IF(ASC(I$)AND127)<32THENWAIT207,1,1:
REM STEUERZEICHEN
470 POKE207,0:PRINTI$;:POKE204,0:REM BLI
NKEN STEUERN
480 GETI$:IFI$="_"THENPOKE204,1:PRINT"██
MOMENT BITTE":GOTO280
490 PRINT#1,I$;:GOTO450:REM ABFRAGESCHLE
IFE
500 REM
510 REM UNTERPROGRAMME
520 GOSUB710:PRINT"    "K$"█"
530 FORI=0TOK%(J):IFJ<>1THENPRINT"█";
540 PRINT"    "K$(I,J):NEXT
550 PRINT"██████████████████████> CURS
OR ODER RETURN █";
560 PRINT"████████";:IFJ<>1THENPRINT
570 IFK(J)=0THEN600
580 FORI=1TOK(J):PRINT"█";:IFJ<>1THENPRI
NT
590 NEXT
600 PRINT"  █"K$(K(J),J)
610 GETK$:IFK$=""THEN610
620 IFK$="□"THENIFK(J)>0THENPRINT"□  "K$
(K(J),J):K(J)=K(J)-1:GOTO560
630 IFK$="█"THENIFK(J)<K%(J)THENPRINT"□
"K$(K(J),J):K(J)=K(J)+1:GOTO560
640 IFK$<>CHR$(13)THEN610
650 RETURN
660 IF(64ANDPEEK(37150))THEN660:REM PUFF
ER LEEREN
670 RETURN
680 KR=128*K(3)+32*K(2)+K(1):BR=32*K(6)+
16*K(5)+K(4):RETURN
690 FORI=0TO7:POKE828+I,K(I):NEXT:RETURN
700 FORI=0TO7:K(I)=PEEK(828+I):NEXT:RETU
RN
710 PRINT"██  RS-232 EIN/AUSGABE  ██":RE
TURN
READY.
```

280 mit der Übernahme der Textblöcke in Stringvariable bis Zeile 300.

In den Zeilen 310 bis 340 werden die Menüs zur Parametereinstellung (Unterprogramm ab Zeile 520) angewählt und die errechneten Registerinhalte (UP 680) angezeigt. Nachdem die gewünschte Einstellung erfolgt ist, wird diese Schleife verlassen und die Testroutine ab Zeile 370 begonnen.

Darin wird zuerst der Bildschirm vorbereitet (UP 710), dann werden die aktuellen Werte von K(I) gerettet (UP 690) und gewartet, bis der Sendepuffer einer eventuell noch laufenden Übertragung ausgeschrieben ist (UP 660). Danach wird das File geschlossen. Dabei wird der RAM-Bereich der beiden Puffer für Basic wieder freigegeben. Weil hierbei ein CLR-Befehl ausgeführt wird, müssen die vorweg geretteten Parameter zurückgeholt (UP 700) und die Registerinhalte berechnet werden (UP 680). Darauf wird das File mit den neuen Parametern eröffnet, der Bereich wieder für Basic gesperrt und die dabei erneut gelöschten Parameter zurückgeholt (UP 700).

In Zeile 390 wird der Puffer initialisiert. Darauf (410 bis 420) wird ein Zeichensatz übertragen und gleichzeitig invertiert auf den Bildschirm geschrieben. In Zeile 430 wird ein weiterer Testsatz gesendet. Zur Vorbereitung des weiteren Tests mit Zeichen von der Tastatur her wird der Cursor auf Blinken geschaltet (440).

In den Zeilen 450 bis 490 liegt die Sende- und Empfangsschleife. Sind alle Zeichen aus dem Empfangspuffer verarbeitet, dann erfolgt gleich der Sprung zur Tastaturabfrage (nach 480). Ein dort eingegebener Rückpfeil beendet den Test und läßt zum Einstellungsmenü verzweigen. Andere Zeichen gehen in den Sendepuffer. Wurde ein Zeichen in Zeile 450 empfangen, dann wird, falls es ein Steuerzeichen ist, gewartet, bis der blinkende Cursor wegschaltet (470), sonst wird das Blinken auch bei aktivem Cursor ausgeschaltet, das Zeichen geschrieben und das Blinken wieder eingeschaltet.

Im Unterprogramm ab Zeile 520 werden sowohl die einzelnen Gruppen als auch die veränderlichen Parameter dieser Gruppen angezeigt, durch Cursor-up- und Cursor-down-Tasten verändert und durch »RETURN« übernommen.

In diesem Testprogramm konnte auf die Statusabfrage der RS232-Schnittstelle verzichtet werden, bei einer Anwendung in einem regulären Treiberprogramm sollte sie aber benutzt werden.

(Hans-Jürgen Kellermann/ev)



Erste Hilfe für den C 64: RENEW

Manchmal läuft auch alles schief! Da waren die Finger wieder einmal schneller als die Gedanken. Man hat NEW eingetippt und erst nach dem Drücken der RETURN-Taste wird einem der angerichtete, ungewollte Schaden klar: Das Programm ist weg! Oder?

Die VC 20-Besitzer können jetzt nur lächeln — wenn sie die Ausgabe 4 kennen. Dort ist nämlich ein Programm abgedruckt, daß ein mit NEW gelöschtes Programm wieder sichtbar macht. Hier ist die angepaßte Version für den C 64.

Eine Programmbeschreibung erübrigt sich, da in der Ausgabe 4 Helmut Welke dies bereits sehr ausführlich gemacht hat. Es sind auch nur sehr kleine Änderungen zu machen. Im Originallisting für den VC 20 (4/84 Seite 89) sind in Zeile 1005 der Wert 207 durch 175 und in Zeile 1015 der Wert 198 durch 166 zu ersetzen und schon läuft's auch im C 64. Bei diesem hilft dieses RENEW auch nach einem Reset oder SYS 64738. Auch dieses Programm kann selbstverständlich mit dem DATA-WANDLER abgespeichert werden.

Listing von »RENEW«

```
100 REM *** RENEW ***
110 FORI=678 TO 755:READA:POKEI,A:NEXT
111 DATA165,43,24,105,4,133,253,165,44,1
05,0,133,254,160,0,177,253,240,8
112 DATA200,192,88,208,247,76,8,175,200,
152,160,0,24,101,253,145,43,133,253
113 DATA144,2,230,254,165,254,200,145,43
,136,177,253,170,200,177,253,240
114 DATA7,133,254,134,253,76,213,2,165,0
,24,105,2,133,45,165,254,32,85,166
115 DATA76,156,166
READY.
```

Datawandler

Mit dem Datawandler ist es auch dem »Nur-Basicprogrammierer« möglich, Maschinensprachprogramme (oder Teile davon), die in Form eines Basic-Laders (also über DATAs und POKE-Befehle) eingetippt wurden, als Maschinensprache abzuspeichern. Dadurch ergibt sich auf der Diskette eine Platz-, beim Laden eine Zeitersparnis.

»Bitte warten — ich lese Daten« — so oder ähnlich wird der »Basicprogrammierer« nach dem Programmstart darauf aufmerksam gemacht, daß bei jedem RUN eines Basicprogramms die Maschinenspracheteile DATA für DATA in die Speicherzellen gePOKEt werden. Warum also nicht die DATAs gleich wieder als Maschinensprache abspeichern! Aber es ergibt sich nicht nur eine Zeitersparnis beim Programmlauf, sondern auch beim Laden, da auf der Diskette oder Kassette weniger Speicherplatz benötigt wird (zum Vergleich: der Schatzsucher aus der Ausgabe 6 belegt auf der Diskette als Basicprogramm 72 Blöcke, als Kombination Basic/Maschinensprache nur noch 44 — außerdem startet dieses Programm dann in Sekundenbruchteilen).

Darüber hinaus bestehen viele Hilfsprogramme, die auf Maschinenroutinen zurückzugreifen, nur aus der Zeile »FORI=xTOxx:READx:POKEI,x:NEXTI« und vielen, vielen DATAs. Derartige Programme bieten sich für den Datawandler von selbst an, da sie nach der Umformung geladen werden können, ohne ein eventuell im Basicspeicher stehendes Programm zu zerstören.

Zum Programm selbst

Vor dem Start des Datawandlers müssen die Daten im Speicher stehen — falls sie nur in Form von DATA-Zeilen vorhanden sind, wird in Zeile 60130 eingefügt:

60130 FORI=AA TO EA:READX:POKEI,X:NEXTI

In den Zeilen 60030 — 60040 wird (in dezimaler Form) die Anfangs- und Endadresse abgefragt, unter der die DATAs »abgelegt« sind und den Variablen AA beziehungsweise EA zugeordnet. Die Zeilen 60050 bis 60100 dienen der Abklärung, ob das abzuspeichernde Programm auch wieder an die Adresse geladen werden soll, wo derzeit die DATA stehen (durch das Verschieben der Ladeadresse ist auch ein leichteres Experimentieren in den Autostartbereichen möglich, die dem Programmierer gelegentlich nach den ersten »POKEs« das Konzept aus der Hand nehmen).

Wird die Abfrage mit »N« beantwortet, so wird nach dieser späteren Ladeabdresse gefragt — lautet die Antwort »J«, so wird die Ladeadresse = derzeitige Anfangsadresse (LA=AA) und nach dem Namen gefragt, unter dem das Programm nun

abgespeichert werden soll (Zeile 60120).
Zeile 60130 — siehe oben.

So dann wird der Floppykanal geöffnet (Zeile 60140), die dezimalen Eingaben auf »Diskettenformat« gebracht (Zeile 60150), damit die ersten beiden Bytes auf der Diskette als Ladeadresse geschrieben werden können (Zeile 60160). In den Zeilen 60170 bis 60190 werden schließlich die Daten ausgelesen und direkt auf die Diskette geschrieben. Die übrigen Zeilen dienen der Abfrage des Fehlerkanals der Floppy beziehungsweise schließlich der »Fertig«-Meldung.

Weitere Hinweise:

1) Das Programm ist in dieser Form sowohl für den VC 20 als auch für den C 64 verwendbar.

2) Bei entsprechender Abänderung des OPEN-Befehls sollte auch die Abspeicherung auf Kassette möglich sein (die Zeilen 60200 bis 60240 entfallen dann).

3) Vor dem Abspeichern sollten die DATAs natürlich korrekt sein, da nach dem Abspeichern eine Überprüfung noch schwieriger ist.

Bei Basicprogrammen sollten deshalb der Basicteil und die DATAs zunächst unabhängig voneinander eingegeben, zum Probelauf mit »MERGE« zusammengefügt und bei Fehlerlosigkeit der DATA-Teil dann entsprechend abgespeichert werden.

4) Die »eigenständigen« Maschinenprogamme werden dann mit LOAD'xy',8,1 geladen und mit dem SYS-Befehl gestartet. Bei Basicprogrammen sollte dann (sofern nicht ein Autostartprogramm zum Laden aller Teile verwendet wird), die erste Programmzeile lauten:

IFA=0THENA=1: LOAD »Name des Maschinensprache-teils«,8,1

— das klingt zwar paradox, aber es funktioniert: nach dem RUN wird dann geladen und gestartet.

Daß in einem solchen Basicprogramm alle READ-Befehle etc. ausgebaut werden müssen, versteht sich wohl von selbst.

(Uwe Christian Parpart/gk)

Listing »Datawandler«

```
60000 REM *** DATAWANDLER ***
60010 REM *** (C) UWE CHR. PARPART ***
60020 PRINT"        *** D A T A W A N D L
E R ***"
60030 INPUT" JETZIGE ANFANGSADRESSE";AA
60040 INPUT"     JETZIGE ENDADRESSE";EA
60050 PRINT" IST DIE ANFANGSADRESSE IDE
NTISCH"
60060 PRINT" MIT SPAETERER LADEADRESSE
(J/N)?"
60070 GETA$:IFA$=""THEN60070
60080 IFA$="J"THENLA=AA:GOTO60120
60090 IFA$="N"THEN60110
60100 GOTO 60070
60110 INPUT"    SPAETERE LADEADRESSE";LA
60120 INPUT" NAME DES PROGRAMMS";L$
60130 OPEN1,8,1,L$+",P,W"
60140 HB=INT(LA/256):LB=LA-HB*256
60150 PRINT#1,CHR$(LB);CHR$(HB);
60160 FORI=AATOEA
60170 PRINT#1,CHR$(PEEK(I));
60180 NEXTI:CLOSE1
60190 REM *** ABFRAGE FEHLERKANAL ***
60200 OPEN1,8,15
60210 INPUT#1,A,B$,C,D
60220 PRINTA;B$;C;D
60230 CLOSE1
60240 PRINT"F E R T I G !"
60250 END
READY.
```

Simons Basic: Befehle, die nicht im Handbuch stehen

Als Ergänzung zu den Artikeln über Simons Basic in den Ausgaben 4/84 und 5/84 wollen wir in dieser Ausgabe noch einige Befehle und Besonderheiten aufführen, die nicht in jedem Handbuch stehen.

Zunächst die zusätzlichen Befehle in alphabetischer Reihenfolge:

BCKGNDS
Syntax: BCKGNDS f1,f2,f3,f4
— f1: normale Hintergrundfarbe
— f2: Hintergrundfarbe der Zeichen mit SHIFT-Taste
— f3: Hintergrundfarbe der REVERS-Zeichen und des Cursors (nicht der Schriftfarbe)
— f4: Hintergrundfarbe für Zeichen mit SHIFT-Taste im REVERS-Mode
Semantik: BCKGNDS legt die Hintergrundfarben fest und schaltet auf ECM (Extended-Color-Mode), dabei werden von jedem Zeichen zwei Bit vom ASCII-Code abgezweigt: es steht somit nicht mehr der gesamte Zeichensatz zur Verfügung.
NRM macht BCKGNDS rückgängig

COLOUR
Syntax: COLOUR, rf, hf
— rf: Rahmenfarbe
— hf: Hintergrundfarbe
Semantik: COLOUR setzt Rahmen- und Hintergrundfarbe und erspart somit das lästige POKE 53280,rf : POKE 53281,hf.

DISABLE
Syntax: DISABLE
Semantik: Setzt ON KEY-Anweisung außer Kraft

GRAPHICS:
Syntax: GRAPHICS
Semantik: Liefert Konstante \$D000 = 53248; Adresse VIC

NRM
Syntax: NRM
Semantik: NRM macht MEM und BCKGNDS rückgängig.

ON KEY
Syntax: ON KEY Stringausdruck, diverse Anweisungen
Semantik: Wird eine Taste gedrückt, die im Stringausdruck des ON KEY-Befehls enthalten ist, so wird in den Anweisungsteil verzweigt. Die Tastatur wird dabei vor jedem Befehl abgefragt. Ein unbedingter Sprung erfolgt, wenn im Stringausdruck eine »eckige Klammer zu« (\$5D) enthalten ist.

RESUME
Syntax: RESUME
Semantik: RESUME funktioniert nur nach ON KEY. Bei RESUME wird das Programm beim ursprünglichen Befehl fortgesetzt. RESUME entspricht somit dem RETURN bei GOSUB.

SOUND
Syntax: SOUND
Semantik: Liefert Konstante $D400 = 53972; Adresse SID

Punkte, die besonders zu beachten sind

AT
ist auch als Zuweisung möglich. Beispiel A$ = AT (Spalte, Zeile) B$. Die Cursorpositionierung erfolgt schon während der Zuweisung.
DUMP
Matritzen werden nicht angezeigt.
NO ERROR, OUT
NO ERROR schaltet nur ON ERROR ab, OUT gibt die Standardfehlermeldung aus.
OLD
Die Variablenwerte gehen verloren.
REPEAT, LOOP, EXEC
Für jede dieser Anweisungen existiert ein eigener Stack, der bis zu fünf Werte aufnehmen kann.
SCRSV, SCRLD, COPY, HRDCPY,
schließen Datei 1.
TRACE
Der TRACE-Befehl funktioniert nicht nach MEM.
Mehr über Simons Basic in: Das Commodore 64-Buch, Band 5.

(Hans Lorenz Schneider/aa)

Die Suche nach den Synthtischen

Das Programm ermöglicht die systematische Suche nach allen vom Betriebssystem unterstützten Steuerzeichen. Man ist nun nicht mehr angewiesen auf zum Teil lückenhafte Tabellen synthetischer Steuerzeichen, sondern kann sich stattdessen selbst auf die Suche begeben.

Von den so sagenumwobenen »synthetischen Steuerzeichen« war in früheren Ausgaben dieser Zeitschrift schon die Rede. Jedoch erhielt der Leser bislang noch kein einigermaßen handfestes »Kochrezept« zur erfolgreichen Suche nach ihnen. Mit dem folgenden Programm soll diese Lücke geschlossen werden.

Es gibt nach Eingabe des gewünschten ASCII-Wertes alle Tastenkombinationen aus, deren Betätigung die Tastaturdecodierroutine dazu veranlaßt, den entsprechenden ASCII-Wert in den Tastaturpuffer zu schreiben.

Es werden allerdings auch solche Kombinationen aufgeführt, deren »ASCII-Werte« zwar in den Decodiertabellen (die ab $EB81 im Betriebssystem beginnen) an entsprechender Stelle aufgeführt sind, jedoch NICHT im Tastaturpuffer erscheinen (zum Beispiel die »2« bei Betätigung der SHIFT-Taste). Der Leser wird diese jedoch schnell von der ersten Gruppe unterscheiden können.

Zum Schluß noch zwei Bemerkungen:

1) Bei gleichzeitigem Erscheinen von zwei oder mehreren Tastenkombinationen für einen bestimmten ASCII-Code kann oft — jedoch nicht immer — eine Kombination gegen eine andere ausgetauscht werden, ohne das Endresultat zu verändern (Beispiel für eine Ausnahme: Sowohl die RUN/STOP-Taste wie auch die CTRL C-Kombination belegen gemäß der Decodiertabellen den ASCII-Wert »3«, jedoch kann mit »CTRL C« kein Programm abgestoppt werden, da für die Abfrage der RUN/STOP-Taste eine gesonderte Routine zuständig ist, die nur diese Taste erkennt).

2) Alle »synthetischen Steuerzeichen«, für deren Erzeugung der Basic-Interpreter zuständig ist, kann dieses Programm nicht erkennen, da es lediglich auf die Tastaturdecodiertabellen im Kernal Bezug nimmt.

Da das Programm lediglich die Tastaturdecodiertabellen des Betriebssystems benötigt (und die dort aufgeführten ASCII-Werte in ein ARRAY einliest) kann es leicht durch entsprechende Abänderung der in Zeile 20 enthaltenen Anfangsadressen in eine auch auf dem VC 20 laufende Version umgeschrieben werden.

(Engin Gülen/gk)

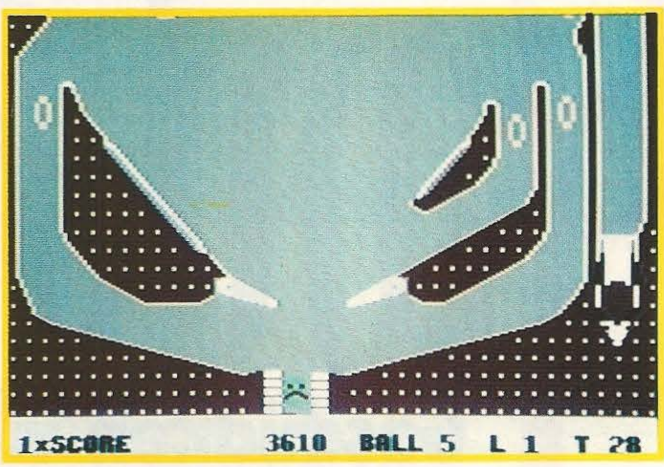
Listing zur Steuerzeichensuche

```
1 REM          *  STEUERZEICHEN   *
2 REM           *  FUER DEN CBM-64 *
3 REM            *  ENGIN  GUELEN   *
4 REM             *  POSTWEG     2   *
5 REM              *  4192 KALKAR 1   *
9 POKE53280,0:POKE53281,0
10 DIMAS(3,63)
18 REM ANFANGSADRESSEEN DER
19 REM TASTATURDEKODIERTABELLEN
20 DATA60289,60354,60419,60536
30 FORI=0TO3:READA(I):NEXT
40 FORI=0TO3:FORJ=0TO63
50 AS(I,J)=PEEK(A(I)+J)
60 NEXTJ,I
159 REM TASTATURMATRIX
160 DATADEL,RETURN,CRSR(RIGHT),F7,F1,F3,F5,CRSR(DOWN)
161 DATA3,W,A,4,Z,S,E,SHIFT(L)
162 DATA5,R,D,6,C,F,T,X
163 DATA7,Y,G,8,B,H,U,V
164 DATA9,I,J,0,M,K,O,N
165 DATA+,P,L,-,.,DOPPELPUNKT,@,KOMMA
166 DATA£,*,;,HOME,SHIFT(R),=,↑,/
167 DATA1,←,CTRL,2,SPACE,C=,Q,RUN/STOP
170 DIMTA$(64):FORI=0TO63:READTA$(I):NEXTI
180 DATA-----,SHIFT,C=    ,CTRL ,
190 FORI=0TO3:READA$(I):NEXT
200 INPUT"████████████ ASCII-KODE ";AS%
210 IFAS%<0ORAS%>255THEN200
220 PRINT:PRINT:FORI=0TO3:FORJ=0TO63
230 IFAS%=AS(I,J)THENPRINT"█"A$(I),:GOTO250
240 GOTO270
250 IFLEFT$(TA$(J),2)=LEFT$(A$(I),2)THENPRINT:GOTO270
260 PRINT"◣"TA$(J)
270 NEXTJ,I
280 PRINTSPC(193)"██ TASTE "
290 GETA$:IFA$=""THEN290
300 GOTO200

READY.
```



Gleich wird die Kugel zerstampft



In der Flipper-Familie ist »Slamball« ein Außenseiter. Sein Spielfeld ist so groß, daß es gar nicht auf einmal auf den Bildschirm paßt.

Wer in seiner Jugend etliche Markstücke in die klingelnden und klappernden Flipperautomaten gesteckt hat, hat sich oft gewünscht, eine dieser Maschinen zu besitzen. Doch leider waren sie zu teuer. Heute geht das einfacher und auch billiger. Auf Diskette oder Kassette ist das Programm »Slamball«, das den Heimcomputer in eine lustige Flippermaschine verwandelt, zu haben.

Daß sich dieser Computerflipper in manchen Dingen doch von einem richtigen Flipper unterscheidet, wird gleich nach dem Laden klar. Das Rollfeld der Kugel ist nicht vollständig abgebildet, sondern läuft zum oberen Rand hin aus. Auch wird nicht mit den Flipperbacken, sondern mit dem Joystick gespielt. Ein immer wieder interessantes Bild: Wie von einer Rakete abgeschossen, jagt der Ball nach dem Start an die Spitze des Flippers. Dabei schiebt sich das gesamte Spielfeld von unten nach oben über den Bildschirm. Damit der Ball, dem ein Gesicht aufgemalt wurde, möglichst lang in Bewegung bleibt, stehen mehrere Paare Flipperbacken zur Verfügung.

Hat man genug Übung, kann ein gewiefter Spieler den Ball mit dem Joystick ähnlich wie bei einem richtigen Flipper manipulieren.

Doch Vorsicht, der »Tilt« funktioniert auch!

Ewig läßt sich die Kugel aber nicht im Spiel halten. Nicht etwa, daß sie einfach aus dem Bild verschwindet. Sie wird, fast brutal, von zwei Schmiedehämmern zerstampft.

Insgesamt ist »Slamball« mehr ein Spiel, als eine echte Flippersimulation. Kaum ein Flipper hat eine lächelnde Kugel, die bei Spielende zerstampft wird oder einen Joystick, um die Flipperbacken zu steuern. Einem jüngeren Spieler wird »Slamball« sicher eine Menge Unterhaltung bieten. Die Älteren wollen wahrscheinlich nicht auf das »hautnahe« Gefühl der rollenden und anschlagenden Kugel eines echten Flippers verzichten.

(Arnd Wängler)

Lock-Ruf: Hacker, meldet Euch

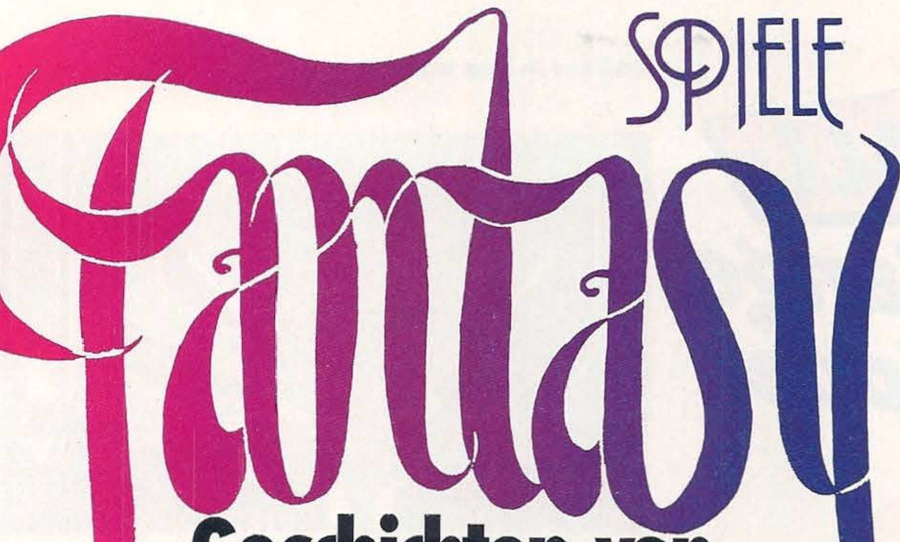
Ist Ihre Telefonrechnung in letzter Zeit sprunghaft angestiegen? Besitzen Sie eine Liste von Telefonnummern, bei denen sich am anderen Ende der Leitung keine menschliche Stimme meldet? Kommen Sie manchmal übernächtigt im Büro oder in der Schule an, obwohl sie nicht durchgefeiert haben? Zeigt man Verständnis, wenn Sie leise vor sich hinträllern: »Mal kommste rein, mal fliegste raus«?

Genau Sie werden von uns gesucht — melden Sie sich doch bei uns. Anruf oder Postkarte genügt.

Für alle, die bis jetzt ratlos geblieben sind: Wir suchen Hacker. Wir wollen wissen, wie das ist — das Stöbern in fremden Datenbanken. Wir suchen Hacker, die die »Szene« kennen. Keine Angst, nach dem Gesetz ist Hacken nicht verboten.

Und jetzt einmal ohne Modem oder Akustikkoppler telefonieren: 089-4613-269; ich freue mich über jeden Anruf — übrigens, ich heiße Karin Gößlinghoff. Wer lieber schreibt, sollte seine Nachricht an folgende Adresse schicken:

Redaktion
Computer persönlich
Karin Gößlinghoff
Hans-Pinsel-Str. 2
8013 Haar bei München



Geschichten von Galaxien, Gnomen und Giganten

Sie heißen Hobbit, Voodoo Castle, Ultima, Enchanter oder Sorcerer. Die Zahl der Fantasy-Rollenspiele wächst täglich, und für manche benötigt selbst der Routinier rund 300 Stunden, um zum Ziel zu kommen.

Ein Androide kämpft auf einem fernen Planeten ums Überleben. Der griechische Sagenheld Odysseus schlägt sich mit Kameraden durch das Mittelmeer. Astronauten entdecken ferne Galaxien, und stolze Helden durchstreifen mystische Welten.

Abenteuer, die jeden Abend über Tausende von Bildschirmen flimmern und in keinem Fernsehprogramm stehen. Die sogenannten Abenteuerspiele oder Adventures und Fantasy-Spiele haben den üblichen Fernseh- und Kinofilmen etwas voraus: Der Zuschauer ist gleichzeitig der Hauptakteur, der Held der Geschichte. Der Computer macht's möglich.

Dabei hatte es vor rund zehn Jahren ohne Computer und Flimmerkiste angefangen. Der Amerikaner Gary Gygax träumte in seiner Jugend (jetzt übrigens auch noch) gerne von Fantasiewelten, was ihn auch dazu brachte, alle klassischen Sagen Europas zu lesen. Als er all diese Sagen gelesen hatte, fand er das Ganze einfach unvollendet: Ihm fehlte die Fortsetzung, die Gary Gygax in Form von Fantasyspielen schreiben wollte. Die ersten »weiterführenden Kapitel« zur Geschichte der Fantasy-Rollenspiele schrieb er schon im Sandkasten. Dort spielte er ein Taktikspiel namens »Chainmail« mit seinen Freunden.

Der nächst Schritt war die Entwicklung des ersten Rollenspiels, »Tunnels and Trolls« (der deutsche Titel ist »Schwerter und Dämonen«). Weitere Rollenspiele sind Runequest, Dungeons& Dragons (die bekannteste Variante), und diverse andere (vom Bereich Science Fiction bis zu James Bond). Später, als die ersten besseren Homecomputer auf den Markt kamen, versuchte man, die in USA sehr populären Rollenspielsysteme auf dem Computer zu realisieren. Teilweise sind diese Versuche mit fesselnden Abenteuerspielen gelungen, aber oft wurden sie auch zu kläglichen Beweisen für die Einfallslosigkeit der Programmierer. Beispiele für gute Rollenspiele am Computer sind zum Beispiel Ultima II, Gateway to Apshai, Exodus, Enchanter, The Hobbit, Blade of Blackpool, und viele andere.

Doch was ist eigentlich ein Rollenspiel? Der Begriff erklärt sich daraus, daß bei dieser Kategorie von Spielen der Spieler die Rolle einer bestimmten Figur übernimmt, den sogenannten »Character«. Durch das Übernehmen des Charakters oder Wesens einer Figur wird man also Zauberer, Abenteurer, Kämpfer, Dieb oder ein ähnliches Wesen. Im Rollenspiel muß man mit den Fähigkeiten dieser Figur (Stärke, Weisheit, Intelligenz, Charisma, Konstitution) und ihren Mitteln (Waffen, Rüstungen, Zaubersprüche, Diebeswerkzeug und ähnliches) eine bestimmte Aufgabe lösen. Bei den Computern kann diese Aufgabe darin bestehen, in den höchsten Level zu kommen (wie bei Gateway to Apshai), oder man muß eine Gegend durchforschen und dabei möglichst viel Erfahrung sammeln. Spielt man die »computerlosen« Rollenspiele, dann hat man zwar in jedem Abenteuer eine Aufgabe (Gegend durchforschen, Feinde bekämpfen, Erfahrung sammeln, Schätze suchen), kann aber mit seinem Character ohne festes Ziel von

So können Hilfsmittel für ein Fantasy-Spiel ohne Computer aussehen: zwei der unzähligen Handbücher... (rechts)

... und so sieht ein Bildschirm bei einem Textabenteuer (in diesem Fall Enchanter; rechts oben) aus.

Abenteuer zu Abenteuer wechseln und beim nächsten Adventure mit seinen vorher erworbenen Erfahrungen, Waffen, Goldmünzen, und so weiterspielen. Solche Rollenspiele sind also endlos (außer, wenn das Wesen , dessen Rolle man übernommen hat, stirbt). Gerade diese Tatsache ist einer der größten Unterschiede zwischen den Rollenspielen mit und ohne Computer. Hat man beispielsweise ein Computer-Adventure zu Ende gespielt, dann ist es aus — für immer.

Die Autoren dieses Berichts spielen selbst gerne Computer-Fantasy, aber auch das computerfreie Spiel »Advanced Dungeons & Dragons«. Die Spielregeln für dieses Spiel, das abgekürzt auch AD&D genannt wird, sind in einem zirka 350 Seiten starken PLAYER'S HANDBOOK zusammengefaßt. Der Leiter eines solchen Spiels hat ein noch weit umfangreicheres, sogenanntes DUNGEON MASTER'S GUIDE als Nachschlagewerk. Das bedeutet natürlich nicht, daß man all dies auswendig lernen muß, um ein solches Spiel spielen zu können, aber diese Zahlen sollen Ihnen nur die Komplexität der Spielregeln und die zahlreichen Möglichkeiten eines AD&D-Spiels zeigen. Die Feinde, auf die man in einem Abenteuer stoßen kann, sind auch vielfältiger Natur: Das MONSTER MANUAL gibt Auskunft über mehr als 350 verschiedene Monster. Das Buch beschreibt beispielsweise, wieviel Schaden man einem Monster zufügen muß, um es zu vernichten, welche geistigen Fähigkeiten das Monster hat, und so weiter.

Was den Spieler selbst betrifft: Seine Fähigkeiten werden ausgewürfelt. Danach kann er sich den Charakter aussuchen. Bei hoher Geschicklichkeit wird er natürlich die Rolle eines Diebes übernehmen, um seine Fähigkeiten optimal ausnutzen zu können. Hohe Weisheit gibt gute Voraussetzungen für die Rolle des sogenannten »Cleric« (weil hohe Weisheit einen Zauberspruchbonus für Kleriker gibt), etc.

Der Spielhergang läßt sich natürlich nicht in Kürze erklären. Es sei hier jedoch gesagt: Wenn ein bestimmter Vorgang in einem Abenteuer nicht vorgesehen ist, kann der Spielleiter (Dungeon Master) selbst noch entscheiden, wie es weitergeht. Bei Computerspielen trifft der Computer leider keine eigenen Entscheidungen.

Die Treffer und deren Stärke beim Kampf gegen Feinde werden ebenfalls ausgewürfelt. Würfel (4seitige, 6-seitige, 8-, 10- und 20-seitige), sogenannte »Module« (die einzelnen Abenteuer), sowie Bücher dazu gibt es im Fachhandel (bessere Spielwarenläden, Fantasy-Büchershops und so weiter). Fantasy-Rollenspiel-Adventures ohne Computer, wie zum Beispiel Dungeons & Dragons, werden von der Firma FSV vertrieben.

Rollenspiele am Computer

Die Rollenspiele auf dem Computer werden von ihren amerikanischen Herstellern »D&D-type-Games« genannt. Ihr großer Nachteil: Sie erlauben dem Spieler nur das, was der jeweilige Programmierer in sein Spiel eingebaut hat.

So ist denn auch kein Wunder, wenn viele Abenteuerspiele nur eine sture Reihenfolge von ganz bestimmten Zügen erlauben. Diese Spiele kann man jedoch nicht mehr unbedingt als Abenteuerspiel bezeichnen, denn hier durchlebt man kein Abenteuer, sondern knobelt meist nur aus, welche Worte der Computer versteht und welche er nicht akzeptiert. Bei vielen gut gelungenen Fantasyspielen fällt dem Spieler jedoch meistens nicht mehr auf, daß er durch die Programmierung eingeschränkt ist. Als Beispiel möchten wir hier die Spiele ULTIMA II und EXODUS angeben: In ihnen sind derartig große und komplexe Welten aufgebaut, daß der Spieler gar nicht mehr merkt, daß irgendwo ein Ende der großen weiten Phantasiewelt besteht.



Arten der Computer-Fantasy

Man kann die Rollenspiele mit Computern in zwei große Gruppen aufteilen: Die joystick- und tastaturgelenkten Spiele (wie KAIV, GATEWAY TO APSHAI, ULTIMA II/III) sowie die textorientierten Spiele, bei denen man wiederum Text/Grafik-Abenteuer und reine Textadventures unterscheiden kann. Die joystick- und tastaturgesteuerten Spiele sehen meist so aus, daß der Spieler seine Figur(en) mit dem Steuerknüppel durch Phantasiewelten, Dungeons, oder »sonstwas« lenkt, wobei er noch zusätzliche Tasten zum Ausführen von Funktionen drücken darf. Diese Funktionen können zum Beispiel sein: Fallen suchen, Zaubersprüche aussprechen, Waffen wechseln, ein Gespräch beginnen, und, und, und...

Die textorientierten Programme akzeptieren Texte als Befehle und reagieren entsprechend. Das Repertoire reicht von primitiven Ein-Wort-Kommando-Adventures bis hin zu komplexen Riesen-Abenteuerspielen, die vollständige Sätze verstehen. Oft sind diese Spiele durch eindrucksvolle Grafik illustriert.

Als kleines Nebenprodukt der Rollenspiele gibt es die sogenannte »Action-Fantasy«. Die Hersteller solcher Spiele nennen sie zwar Fantasy-Spiele, aber eigentlich handelt es sich nur um reine Actionspiele, die durch ein Fantasy-Thema aufgereichert wurden, wie zum Beispiel das Commodore-Spiel DRAGONSDEN.

Hier nun eine kleine Marktübersicht über Fantasy-Rollenspiel-Adventures auf dem C 64:

Fantasy auf dem C 64

Wenden wir uns erst den joystick- und tastaturgelenkten Spielen zu:

ULTIMA II:

Dieses Programm ist eines der umfangreichsten Spiele, die es für den C 64 gibt. Ziel des Spieles ist es, den bösen »Minax« zu töten. Um dies zu schaffen, muß man viel Zeit und Geduld haben und etliche Kämpfe durchstehen.

Tastaturgesteuert (fast alle Tasten sind belegt).

Herstellerfirma: Sierra-On-Line

GATEWAY TO APSHAI:

Bei diesem Spielt heißt es, den höchstmöglichen Level zu erreichen. Dies gelingt nur mit viel Erfahrung, die man in diversen joystickgelenkten Kämpfen sammelt.

Hersteller: Epyx.

Die WARRIORS OF RAS-Serie:

In dieser Serie erschienen Spiele wie KAIV, DUNZHIN, und so weiter. Sie sind jedoch nur dem zu empfehlen, der schon einmal Fantasy-Rollenspiele (ohne Computer) gespielt und viel Zeit und vor allem Geduld hat. Auf Grafik wird in diesen Spielen nicht viel Wert gelegt, das Spielerische dominiert.

Herstellerfirma: Screenplay.

TELENGARD:

Dieses Realtime-Abenteuer ist nur dem Profi zu empfehlen. Man muß sehr schnell denken, um im richtigen Augenblick die richtige Taste zu drücken. Beim Durchforschen der Dungeons stößt man oft auf Drachen und ähnliche Wesen, die sehr schwer zu besiegen sind.

Herstellerfirma: Avalon Hill.

EXODUS - ULTIMA III:

Exodus ähnelt dem Spiel »Dungeons and Dragons« wohl am meisten. Ziel ist es, Exodus zu finden. Ein Test dieses Spiels erfolgt eventuell in Ausgabe 11 von 64'er. Zu EXODUS werden sehr umfangreiche Anleitungen mitgeliefert, so beispielsweise eine auf Stoff gedruckte (farbige) Landkarte, ein Zauberspruchbuch, eine Spielanleitung etc., zum Teil auf alten Pergamentschriftrollen gedruckt. Die durchschnittliche Spielzeit bis zur endgültigen Lösung des Rätsels dürfte bei 200 bis 300 Stunden liegen.

Herstellerfirma: Origin Systems.

Schreiten wir nun zu den textorientierten Fantasy-Spielen, zu den sogenannten »Adventures«:

Reine Textadventures

INFOCOM-ADVENTURES:

Diese Abenteuerspiele sind wohl die besten Textadventures, die es derzeit auf dem Markt gibt. Sie haben einen Wortschatz von über 800 Worten und verstehen ganze Sätze (wie zum Beispiel »Go to the north, then look around and open the green door«). Das Spektrum der Spielthemen geht vom Mittelalter über Detektivspiel bis hin zu Science Fiction oder »20000 Meilen unter dem Meer«-Abenteuern. Die Spiele werden in pompösen Packungen geliefert: Der Inhalt besteht je nach Spiel aus altertümlichen Schriftrollen, Postkarten aus der Galaxis, kleinen Identifikations-Magnetkarten, Spielbrettern, Karten mit Geheimschrift, Weltraumkarten, und, und, und... Die Infocom-Abenteuer sind trotz des relativ hohen Preises sehr empfehlenswert. Spielnamen: WITNESS, DEADLINE, ENCHANTER, SUSPENDED, ZORK I/II/III, PLANETFALL, SORCERER und SEASTALKER.

Herstellerfirma: Infocom.

CYBORG:

Hier ist man der menschliche Teil eines Wesens, das halb Roboter, halb Mensch ist. Um seine Abenteuer zu bestehen, muß man sich mit dem Robotergehirn koordinieren. CYBORG versteht ganze englische Sätze.

Grafik-Abenteuer

Sirius-Adventures:

BLADE OF BLACKPOOL hat altertümliche Komponenten; Ziel des Spiels ist es, das »Schwert des Blackpool« zu finden. CRITICAL MASS liegt mehr in der Gegenwart: Man muß die Atombomben finden, die ein Bösewicht in fünf großen Weltstädten versteckt hat. GRUDS IN SPACE bezieht sich mehr auf die Zukunft. Das Ziel des Spiels erscheint auf einem Bildschirm, den man durch die Eingabe »press green button« aktiviert. Diese Spiele begeistern vor allem durch ihren Einfallsreichtum, ihre sehr gute Grafik und durch ihren Spielwitz.

Herstellerfirma: Sirius-Software.

Die Scott-Adams-Abenteuer:

Diese »Adventure Games«, die es schon auf dem VC 20 als Textversion gab, gibt es jetzt für den Commodore 64 in Grafikversion. Der Text ist etwas knapper gehalten als in der reinen Textversion, was aber nicht weiter stört, weil die Grafik (die übrigens klein aber oho ist) die Umgebung ja auch umschreibt. Es handelt sich um VOODOO CASTLE, PIRATE ADVENTURE, MISSION IMPOSSIBLE und THE COUNT.

Außerdem hat die Firma Adventure International die Rechte zu einigen Superhelden-Stories aus dem amerikanischen Marvel-Comic-Imperium aufgekauft. Von diesen Marvel-Comic-Adventures ist bereits das Spiel »THE INCREDIBLE HULK« erhältlich (Grafik sehr gut, Spiel zu schwer). Spiderman und diverse andere sollen noch folgen.

Hersteller: Adventure-International.

THE HOBBIT:

Es geht darum, einen von einem Drachen bewachten Schatz zurückzubringen. Die Story entspricht im Großen und Ganzen dem Buch »The Hobbit« von Tolkien. Das besondere an diesem Abenteuer ist die soge-

nannte »Animaction« (so nennt es der Hersteller), das heißt, daß die Wesen, die in diesem Abenteuer vorkommen, sich bei jedem Spiel anders verhalten, und man sich mit ihnen (»Talk to Name Text«) arrangieren muß, um das Adventure zu lösen. Sehr schwer zu spielen, aber sehr unterhaltsam.

Hersteller: Melbourne House.

DALLAS-QUEST:

Ein Spiel, das für fortgeschrittene Abenteuerer nicht geeignet ist, da es recht schnell zu lösen ist. Dieses Abenteuerspiel bietet jedoch die beste Grafik, die bisher auf einem Commodore 64-Adventure zu sehen war. Sehr überzeugend ist auch der Reichtum an humorvollen Ideen. Dieses Spiel ist also für einen Anfänger-Abenteuerer sehr zu empfehlen.

Hersteller: Datasoft.

Abenteuerspiele von Sierra-On-Line:

Die Spiele MISSION ASTEROID und WIZARD & PRINCESS sind ebenfalls als Anfänger-Abenteuer zu bezeichnen. Sie haben zwar einen kleinen Wortschatz, können aber nach einiger Übung auch von ungeübten Spielern gelöst werden. Ein umfangreicheres Grafikadventure ist »Ulysses and the golden Fleece«, in dem man die Rolle von Odysseus übernehmen muß. Nach einiger Anlaufzeit macht dieses Spiel sehr viel Spaß.

Natürlich können wir in dieser Übersicht nicht alle Fantasy-Spiele beschreiben. Aber wir hoffen, Ihnen einen kleinen Überblick über die bestehende Computer-Fantasy gegeben zu haben. Die versprochenen Lösungshinweise zu »Blade of Blackpool« und »The Hobbit« werden in der nächsten Ausgabe gebracht.

(M. Kohlen/F. Wlodarczyk)



Anmerkungen

Wer sich für Rollenspiele (ohne Computer) interessiert, wendet sich an Ferdinand Wlodarczyk, Prager Str. 26, 8000 München 45.

Rollenspielsets (zum Beispiel D&D-Basis-Set) gibt es in besseren Spielwarenhandlungen zu kaufen.

Computer-Fantasy-Spiele gibt es natürlich in Computerläden und im Versandhandel.

SCHNELLBOOT

Rettung aus der grünen Hölle

Eine typische Szene aus »Schnellboot«

Sie sind Commander eines technisch voll ausgerüsteten Schnellbootes und haben eine gefährliche Mission zu erfüllen. Ihre Aufgabe ist es, drei Forscher zu retten, die im Dschungel verschollen sind. Doch dies ist gar nicht so einfach, denn in den Gewässern befinden sich gefährliche Krokodile und Klippen, die man abschießen kann. Bei diesem Spiel wäre es übrigens angebracht, den Feuerknopf des Joysticks festzuklemmen, da sich nach kurzer Zeit Schmerzen in der Daumengegend bemerkbar machen.

Zum Spiel: Sie führen Ihr Boot stromaufwärts, dabei können Sie den Hindernissen ausweichen oder diese abschießen. An drei verschiedenen Anlegestellen warten möglicherweise die verschollenen Forscher auf Sie. Wann und wo, das erfahren Sie, wenn Sie deren SOS-Ruf empfangen. Sobald alle drei Forscher an Bord sind, fahren Sie zum letzten Steg und kassieren Ihren Bonus als Belohnung.

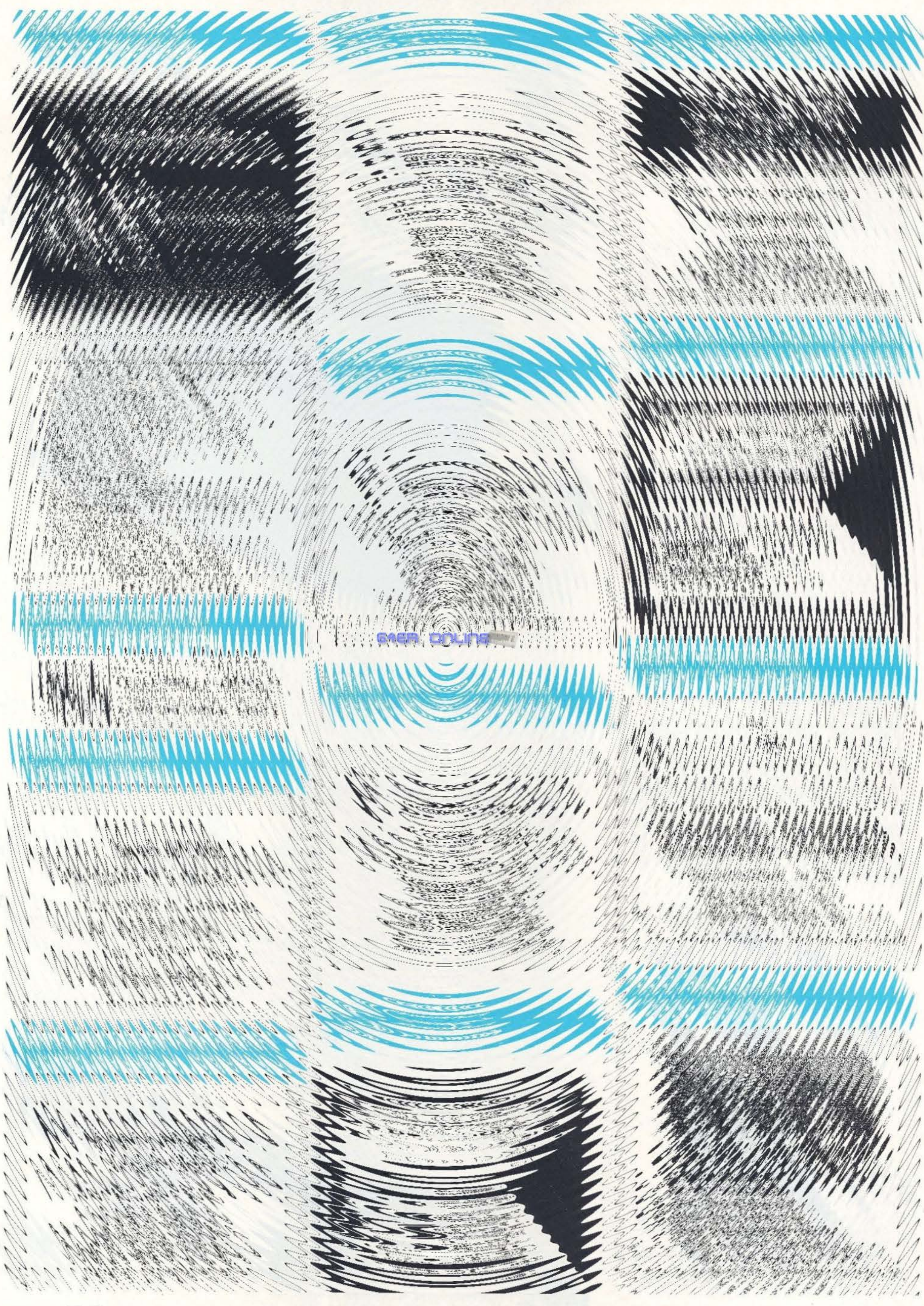
Wenn Sie ein Fan von Schießspielen sind, dann wird Ihnen dieses Spiel sicher eine Weile (!?!) Spaß bereiten.

Gesamtüberblick

Idee	befriedigend
Grafik	befriedigend
Sound	gut
Schnelligkeit	sehr gut
andauernde Spielmotivation	befriedigend
Gesamturteil	gut - befriedigend

»Schnellboot« von THORN EMI VIDEO ist als Steckmodul für den VC 20 erhältlich.

(C.Q. Spitzner/B. Carli)



Das macht den


Bild 1. Die 64-KByte-Karte für den VC 20 von Kalawsky

Eine Frage, die sich wohl ein jeder stellt: Kann ich eine 64-KByte-RAM-Karte überhaupt voll ausnutzen oder reicht schon eine 16-KByte- oder 32-KByte-Karte? Es ist bekannt, daß der VC 20 maximal 27,5 KByte Basicspeicher adressieren kann. Weiterhin können noch 8 KByte ($A000-$BFFF) für Maschinenprogramme oder Spielmodule adressiert werden. So wird hier maximal die Hälfte der 64-KByte-Karte genutzt. Wofür braucht man nun die anderen 32 KByte? Dieser Speicherplatz bleibt natürlich nicht ungenutzt. Er kann je nach Bedarf zum Speichern von Daten verwendet werden. Zuvor ein paar Bemerkungen zum VC 20-RAM-Speicher. In der VC 20-Grundversion ist der Adreßbereich von $0400 bis $0FFF für eine 3-KByte-RAM-Erweiterung vorgesehen. Der Bereich $2000-$7FFF bietet Platz für 24 KByte. Diesen Bereich kann man in Blöcken zu je 8 KByte unterteilen:

1) $2000-$3FFF
2) $4000-$5FFF
3) $6000-$7FFF

Für Toolkits, Erweiterungen von Betriebssystemen und Modulspiele ist der Adreßbereich von $A000-$BFFF reserviert. Mit Hilfe von Dil-Schaltern (Dil bedeutet Dual in line) auf beiden 64-KByte-Karten lassen sich beliebige Speicherkombinationen einstellen:

1) Grundversion
2) 3 KByte RAM-Erweiterung
3) 8 KByte RAM-Erweiterung
4) 16 KByte RAM-Erweiterung
5) 24 KByte RAM-Erweiterung

Zu diesen fünf Möglichkeiten kann jeweils noch der Modulbereich belegt werden.

Die Dil-Schalter sind auf beiden Platinen (in Bild 1 die von Kalawsky, in Bild 2 die von Roos Elektronik) so ungünstig angebracht, daß diese beim Einstecken in den Hauptspeichererweiterungs-Slot des VC 20 ganz im Gerät verschwinden. So muß jedesmal beim Umschalten die Karte aus dem Gerät gezogen werden. Um den anderen Teil der 64-KByte-RAM-Karte zu nutzen, wird bei dem Modul von Kalawsky die Möglichkeit des »Banking« eingesetzt. Durch die Einteilung der 64 KByte in acht Blöcke mit je 8 KByte, die man per Software (POKE I/O 2,X => POKE 39000,X) auswählen kann, ist es tatsächlich möglich, 64 KByte für den VC 20 nutzbar zu machen. Falls I/O 2 schon für zum Beispiel einen IEC-Bus oder eine 40/80-Zeichenkarte benötigt wird, so kann der RAM-Master durch Löten einer Zinnbrücke auf I/O 3 gelegt werden. Die POKE-Adresse ändert sich dann von 39000 auf 40000. »Banking« kann in den Bereichen $6000-$7FFF oder $A000-$BFFF realisiert werden. So können maximal 8 Banks durch Umschalten genutzt werden. Die gesamte 64-KByte-RAM-Karte läßt sich zusätzlich noch durch einen Schreibschutz sichern. Mit einer externen Stromversorgung der Karte kann man dann sogar noch nach Ausschalten des Computers Programme, die sich im Speicher der RAM-Karte befinden, erhalten. Weiterhin kann man damit ein unbeabsichtigtes Überschreiben des Speichers vermeiden.

Wer eine Modulbox besitzt, kann bei dieser Karte sogar seine eventuell vorhandenen Speichererweiterungen wie zum Beispiel 3-KByte-Karte, Superexpander oder 16-

Bild 3. Ein Vergleich der beiden Karten

	Hans Kalawsky	Roos Elektronik
Beschreibung:	dürftig	dürftig
mögliche Ausbaustufen:	alle	alle
Datensicherung nach Ausschalten:	ja	nein
Schalterposition	unzugänglich	unzugänglich
Softwaremäßige Datensicherung:	gut	sehr gut
Preis:	298,- DM	279,- DM
Gesamturteil:	**gut**	**gut**

Die 64-KByte-RAM-Karte von Hans Kalawsky ist auch als Bausatz zu 248 Mark erhältlich. Die Platine ohne Bauteile kostet 42 Mark.

»Kleinen« größer

Wer mit dem minimalen Speicher des VC 20 (3,5 KByte) nicht zufrieden ist, wird sich früher oder später eine Speichererweiterung zulegen. Unter der großen Auswahl von solchen Erweiterungen gibt es seit einiger Zeit auch 64-KByte-RAM-Karten.

KByte-Karte verwenden, falls er die 64-KByte-Karte ausschließlich für Banking benutzt. Es können auch zwei 64-KByte-RAM-Master-Karten verwendet werden.

Zusammenfassung der Anwendungsmöglichkeiten:

1) 64-KByte-RAM durch Banking im Bereich $6000-$7FFF beziehungsweise $A000-$BFFF
2) Vollausbau des VC 20 und Banking
3) Mischung aus 1. und 2., falls weitere Erweiterungen vorhanden sind.
4) Kombination von zwei 64-KByte-RAM-Master-Karten.

Weiterhin ist bei Kalawsky ein Programm erhältlich, das die 64-KByte-RAM-Master-Karte voll ausnutzt und die Banking-Eigenschaften anschaulich vorführt: »Bankingshow«

ROOS Elektronik benützt hingegen eine andere Art der Datenspeicherung für ihre 64-KByte-RAM-Karte. Die Karte ist mit einem nützlichen EPROM bestückt. Die mitgelieferte Software im EPROM muß vor Gebrauch erst einmal mit SYS 43008 aktiviert werden. Durch diese Software gelangen die folgenden sechs Basicbefehle zu einer größeren Bedeutung: OPEN, PRINT#, INPUT#, GET#, SAVE und LOAD.

Man erhält zusätzlich acht Gerätenummern (DEVICE NUMBER 200 bis 207), die in Verbindung mit den erwähnten sechs Basicbefehlen einen superschnellen Datenaustausch zwischen RAM-Speicher und 64-KByte-RAM-Karte ermöglichen.

Ein weiterer Vorteil bei der mitgelieferten Software aus dem EPROM ist, daß kein Filename angegeben werden muß (aber kann). Die Files unterscheiden sich ja durch die verschiedenen Gerätenummern (200 bis 207). Bei sequentiellen Files muß nach OPEN... und PRINT#... ein CLOSE... folgen. Erst damit gehen die zu diesem Zeitpunkt im Puffer stehenden Zeichen nicht verloren. Für den Fall, daß der Speicher der 64-KByte-RAM-Karte voll ist, gibt der Computer die Fehlermeldung »TOO MANY FILES« aus. Die 64-KByte-RAM-Karte von ROOS Elektronik hat wohl zwei mehr oder weniger große Nachteile im Vergleich zur 64-KByte-RAM-Karte von Kalawsky:

Das EPROM benutzt den Adreßbereich von $A000-$AFFF, während die Karte von Kalawsky keinen Speicherbereich des VC 20 belegt. Weiterhin dürfen bei der Roos Elektronik-Karte vorläufig keine anderen Speichererweiterungen (Superexpander, 16-KByte-Karte und so weiter) benutzt werden. Die Karte von ROOS Elektronik wird dafür nach Angaben des Herstellers demnächst in eine Box eingebaut. Auch der Dil-Schalter soll dann an einer zugänglicheren Stelle plaziert sein.

40/80-Zeichenkarte oder IEC-Bus können bei beiden 64-RAM-Karten verwendet werden. In Bild 3 sind die beiden Karten gegenübergestellt.

Beide 64-KByte-RAM-Karten sind wohl, je nach Anwendungsgebiet, gleich gut. Wer sich aufgrund dieses Tests eine dieser 64-KByte-RAM-Karten zulegen möchte, muß sich wohl in erster Linie im klaren sein, ob er Speicherverwaltung durch »Banking« oder durch erweiterte Basicbefehle bevorzugt. Der Sinn von 64-KByte-RAM-Karten für den VC 20 liegt wohl in der Verwendung als Pseudo-Floppy. Dabei lassen sich beispielsweise Daten einer Adreßverwaltung von der Floppy oder der Datasette in diese Pseudo-Floppy einlesen und wesentlich schneller abarbeiten. Nach den durchgeführten Arbeitsvorgängen wie Ändern, Einfügen oder Sortieren werden diese Daten wieder auf die physikalischen Speichermedien transferiert. Das führt zu einer, oft nicht unerheblichen, Zeitersparnis.

(Christian Quirin Spitzner/aa)

Bild 2. Die 64-KByte-Karte für den VC 20 von Roos Elektronik

Ihr Akustikkoppler wird zum

Die Anregung zu der Entwicklung der Schaltung entstand aus dem Ärgernis, daß jeder Besitzer eines Akustikkopplers, wenn er eine Mailbox betreiben will, bei jedem Anruf aufspringen und dann die Anfrage an den Computer weiterleiten muß. Mit Hilfe meiner Schaltung ist es nun möglich, daß der Mailbox-Computer auf einen Anruf reagiert und selbständig die Verbindung herstellt.

Bei einem Anruf wird von der Vermittlungsstelle ein Rufwechselstrom zum Teilnehmer gesendet. Dieser Strom steuert die Leuchtdiode des Optokopplers und der Kopplertransistor wird niederohmig und zieht den BC 547 nach Masse, so daß dieser im Takt des Rufwechselstroms den Eingang (Port H) des C 64 ebenfalls taktet. Wird nun der Port getaktet, so reagiert der Computer und der Pegel am Ausgang (Port F) wird auf 5 V (High) gesetzt. Durch die positive Spannung leitet nun der Relaistransistor, das Relais zieht an und stellt die Verbindung zum Fernsprechapparat her. Nun kann der Computer mit der Datenkommunikation über den Koppler beginnen.

Aufbau der Hardware

Die Schaltung kann problemlos auf einer Experimentierkarte aufgebaut werden. Wichtig ist nur die galvanische Trennung von Fernsprechleitung und Computer-Hardware.

Die Bauteile sind handelsüblich und meist sogar in jeder Bestelliste vorhanden. Wichtig für die Funktion der Schaltung ist nur die Tatsache, daß das Relais bei 5 V sicher schaltet und einen Strom von weniger als 50 mA zieht.

Die Schaltung kann anschlußfertig mit Mailbox-Software beim Autor bezogen werden.

Die DIN-Bezeichnungen entsprechen dem Günther-Dil-Relais 3570 1301 051. Der Anschluß der Schaltung an den C 64 erfolgt über den Userport. Der Koppler wird über den RS232-Stecker (25polig) angeschlossen. Der Anschluß ans Telefonnetz erfolgt folgendermaßen:

Die Leitung a (La) wird von der Anschlußdose getrennt und auf den Anschluß 7 des Interfaces gelegt. Der Anschluß 4 wird parallel auf die Lb aufgelegt. Anschluß 7 wird auf den Anschluß La der Dose geklemmt.

Für die weitere Benutzung des Apparates sollte man parallel zu den Relaiskontakten (1,7) einen Schalter legen, der diese Kontakte überbrückt.

Funktionen

1. Umsetzung des Rufsignals 60 V 25 Hz auf TTL-Pegel zur Auswertung durch den Computer (Auswertung des Rufs durch Computer)
2. V.24-5-Schnittstelle für Akustikkoppler (Datenkommunikation)
3. Anschalten des Fernsprechapparates durch Signal vom Computer (automatisches Abheben)
4. Wahlmöglichkeit für den Benutzer (automatische Wahl)

Bedienungsanleitung

Für die Benutzung der Funktionen 3 bis 4 ist der Schalter in die Stellung »Modem« zu bringen und danach das entsprechende Programm in den Computer zu laden, das heißt, wenn nur die Wahlfunktion genutzt wird, zum Beispiel das Programm Wählautomat. Für den Betrieb einer Mailbox das entsprechende Mailbox-Programm. Der weitere Ablauf ist im Programm beschrieben. Die Funktionen 1 bis 2 sind unabhängig von der Schalterstellung.

Achtung! Diese Schaltung darf nur an Hausanlagen angeschlossen werden, die nicht mit dem öffentlichen Telefonnetz verbunden sind!

Zu Beginn fragt das Programm, ob der Apparat an einer Nebenstellenanlage angeschlossen ist. Ist dies der Fall, so geben Sie bitte »j« ein

Stückliste

Halbleiter
3 x BC 237 B o.ä.
1 x BC 547 B o.ä.
1 x 1 N 4148
1 x LED 5 mm
1 x Optokoppler IL 74
1 x Z-Diode 12 V 400 MW
1 x 1 N 4001

Widerstände Kondensatoren
1 x 150 Ω ¼ W 10%
1 x 330 Ω ¼ W
2 x 2k2 ¼ W
2 x 4k7 ¼ W
1 x 5k6 ¼ W
1 x 10k ¼ W
1 x 56k ¼ W
1 x 1 µF 100 V UK11
1 x 470 µF 16 V Elko

Sonstiges
1 x Dil-Relais, Günther 3570 1301 051
1 x RS232-Stecker (25pol)
1 x Userport-Stecker
1 x Schalter (1 x Um)

Modem und der C 64 zur Mailbox

und geben dann anschließend die Nummer für die Anwahl einer Amtsleitung ein. Diese Nummer wird dann der Telefon-Rufnummer vorangestellt.

Das Programm »Wählautomat«

Nach RETURN gelangen Sie ins Hauptmenü. Mit den Tasten f 1 und f 3 wählen Sie den Menüpunkt an. Ist er gefunden, so drücken Sie bitte RETURN.

Neue Adressen

Geben Sie bitte in die Eingabemaske die entsprechenden Angaben ein und drücken RETURN. Danach gehen Sie in die Zeile mit dem einzelnen Strich und drücken RETURN. Falls Sie noch etwas ändern wollen, so geben Sie bei den Sicherheitsabfragen »n« ein.

Adresse ändern oder löschen

Nachdem Sie mit dem Cursor und den Tasten f 1 und f 3 die gewünschte Adresse gefunden haben, drücken Sie f 5.

Danach können Sie auswählen, ob Sie ändern, löschen oder wieder ins Hauptmenü wollen. Wenn Sie die Daten ändern, gehen Sie bitte nach dem Abschnitt »Neue Adressen« vor. Vor dem Löschen von Daten werden Sie jeweils zur Sicherheit gefragt, ob Sie diese wirklich löschen wollen.

Adresse anwählen

Hier können Sie sich Ihren Gesprächspartner mit den Tasten f 1 und f 3 aussuchen. Danach drücken Sie f 5. Darauf erscheinen die Daten des Teilnehmers und ein weiteres Menü. Drücken Sie bitte gemäß Ihres Wunsches die revers angezeigte Taste. Danach gehen Sie bitte entsprechend dem Menü vor. Für eine Direktwahl drücken Sie f 7.

Der Anschluß

1. extern

Die ankommende Leitung a wird mit dem Anschluß 1 (grau gekennzeichnet) verbunden. Die Leitung a des Telefons wird mit dem Anschluß 2 (isoliert ohne Kennzeichnung) verbunden. Anschluß 3 (Abschirmung) wird parallel auf die Leitung b geklemmt.

2. intern im Apparat: (dazu Deckel abschrauben)

Stecker des Nummernschalters oder Tastwahlblockes abziehen. Anschluß 1 (grau gekennzeichnet) mit Anschluß 1 der 4polige Buchse verbinden. 2 (isoliert ohne Kennzeichnung) mit 2 der Buchse. Abschirmung (Anschluß 3) mit 4 der Buchse. Deckel wieder zuschrauben.

(Rudolf Möllenbeck/rg)

Mailbox-Listings

Listing 1 ist ein Editierprogramm zur Katalogeingabe für die Mailbox.

```
100 REM ++++++++++++++++++++++++++++++++
105 REM +++                          +++
110 REM +++      C A T A L O G       +++
120 REM +++                          +++
130 REM +++  ALL COPYRIGHTS BY       +++
140 REM +++                          +++
150 REM +++      THOMAS KOETHER      +++
160 REM +++                          +++
170 REM ++++++++++++++++++++++++++++++++
1000 :
1010 REM             M E N U E
1020 :
1030 PRINT"▯
1040 PRINT"▯▯▯▯                C A T A L O
G"
1045 PRINT"                ---------------▯▯
"
1050 PRINT"NEUEN CATALOG ANLEGEN........
.....<1>"
1060 PRINT"NEUE PRODUKTE ANHAENGEN......
.....<2>"
1080 PRINT"E N D E......................
.....<0>
1090 PRINT"▯I H R E  E I N G A B E......
......<";
1100 GETW$:IFW$=""THEN1100
1105 PRINTW$
1110 IFW$="1"THEN1185
1120 IFW$="2"THEN1250
1140 IFW$="0"THENEND
1150 PRINT"▯▯▯":W$="":GOTO1090
1160 :
1170 REM       DATEIINHALT LOESCHEN
1180 :
1185 IFG=1THEN1000
1190 OPEN1,8,15,"S:CATALOG":CLOSE1
1200 G=1
1210 GOTO1430
1220 :
1230 REM              INFOS ANHAENGEN
1240 :
1250 IFG=1THENPRINT"DATEI NICHT VORHANDE
N":FORA=1TO2000:NEXT:GOTO1000
1260 OPEN1,8,3,"CATALOG,S,A"
1270 GOSUB9000
1280 GOSUB8000
1290 GETW$:IFW$=""THEN1290
1300 IFW$="Y"ORW$="J"THENPRINT#1,"♠♠♠":G
OTO1270
1310 IFW$="N"THENPRINT#1,"♠♠♠":CLOSE1:GO
TO1000
1320 GOTO1290
1400 :
1410 REM             INFOS EINGEBEN
1420 :
1430 OPEN1,8,4,"CATALOG,S,W":G=0:PRINT#1
,"FIRST"
1440 GOSUB9000
1445 GOSUB8000
1450 GETW$:IFW$=""THEN1450
1460 IFW$="Y"ORW$="J"THENPRINT#1,"♠♠♠":G
OTO1440
1470 IFW$="N"THENPRINT#1,"♠♠♠":CLOSE1:GO
TO1000
1480 GOTO1450
8000 PRINT"▯▯▯▯▯▯▯▯▯▯":X=1
8010 Z$=""
8020 IFX>9THEN8090
8030 GETQ$:IFQ$=""THENPRINT"▯ ▯";:GOTO8
035
8032 GOTO8039
8035 GETQ$:IFQ$=""THENPRINT" ▯";:GOTO803
0
8039 PRINT" ▯";
8040 IFQ$=CHR$(13)THEN8060
8045 IFQ$="_"THENFORA=XTO12:PRINT"▯":NEX
T:GOTO8090
8047 IFQ$=CHR$(20)THENL=LEN(Z$):Z$=LEFT$
(Z$,L-1):PRINTCHR$(157);:GOTO8030
8050 Z$=Z$+Q$:PRINTQ$;:GOTO8030
8060 X=X+1
8070 PRINT#1,Z$:PRINT:GOTO8010
8080 W$=""
8090 PRINT"▯▯▯WOLLEN SIE NOCH MEHR INFOS
 EINGEBEN ?"
8100 RETURN
8110 :
8120 :
8130 :
9000 PRINT"▯▯▯▯▯"
9010 IFG=1THENPRINT"DATEI NICHT VORHANDE
N":FORA=1TO2000:NEXT:GOTO1000
9020 FORA=1TO9:FORB=1TO39:PRINTCHR$(46);
:NEXT:PRINT:NEXT
9030 RETURN
READY.
```

Listing 2 ist ein Editierprogramm zur Informationseingabe in die Mailbox

```
100 REM ++++++++++++++++++++++++++++++++
105 REM +++                          +++
110 REM +++  I N F O S  I N P U T    +++
120 REM +++                          +++
130 REM +++  ALL COPYRIGHTS BY       +++
140 REM +++                          +++
150 REM +++      THOMAS KOETHER      +++
160 REM +++                          +++
170 REM ++++++++++++++++++++++++++++++++
1000 :
1010 REM             M E N U E
1020 :
1030 PRINT"▯
1040 PRINT"▯▯▯▯        A K T U E L L E  I
N F O S
1045 PRINT"        -----------------------
----▯▯▯▯▯▯▯▯▯▯▯"
1050 PRINT"DATEIINHALT LOESCHEN.........
.....<1>"
1060 PRINT"NEUE INFOS ANHAENGEN.........
.....<2>"
1070 PRINT"INFOS IN GELOESCHTE DATEI....
.....<3>
1075 PRINT"INFOS LESEN..................
.....<4>
```

Ihr Akustikkoppler wird zum Modem und

Fortsetzung Listing 2

```
1080 PRINT"E N D E......................
.....<0>
1090 PRINT"▒I H R E  E I N G A B E......
.......<";
1100 GETW$:IFW$=""THEN1100
1105 PRINTW$
1110 IFW$="1"THEN1185
1120 IFW$="2"THEN1250
1130 IFW$="3"THEN1430
1135 IFW$="4"THEN10000
1140 IFW$="0"THENEND
1150 PRINT"▯▯▯":W$="":GOTO1090
1160 :
1170 REM        DATEIINHALT LOESCHEN
1180 :
1185 IFG=1THEN1000
1190 OPEN1,8,15,"S:AKTUELL":CLOSE1
1200 G=1
1210 GOTO1000
1220 :
1230 REM                INFOS ANHAENGEN
1240 :
1250 IFG=1THENPRINT"DATEI NICHT VORHANDE
N":FORA=1TO2000:NEXT:GOTO1000
1260 OPEN1,8,3,"AKTUELL,S,A"
1270 GOSUB9000
1280 GOSUB8000
1290 GETW$:IFW$=""THEN1290
1300 IFW$="Y"ORW$="J"THENPRINT#1,"♠♠♠":G
OTO1270
1310 IFW$="N"THENPRINT#1,"♠♠♠":CLOSE1:GO
TO1000
1320 GOTO1290
1400 :
1410 REM             INFOS EINGEBEN
1420 :
1430 OPEN1,8,4,"AKTUELL,S,W":G=0:PRINT#1
,"FIRST"
1440 GOSUB9000
1445 GOSUB8000
1450 GETW$:IFW$=""THEN1450
1460 IFW$="Y"ORW$="J"THENPRINT#1,"♠♠♠":G
OTO1440
1470 IFW$="N"THENPRINT#1,"♠♠♠":CLOSE1:GO
TO1000
1480 GOTO1450
8000 PRINT"▯▯▯▯▯▯▯▯▯▯▯▯▯":X=1
8010 Z$=""
8020 IFX>12THEN8090
8030 GETQ$:IFQ$=""THENPRINT"▒ ▒▒";:GOTO8
035
8032 GOTO8039
8035 GETQ$:IFQ$=""THENPRINT" ▒";:GOTO803
0
8039 PRINT" ▒";
8040 IFQ$=CHR$(13)THEN8060
8045 IFQ$="_"THENFORA=XTO12:PRINT"▒":NEX
T:GOTO8090
8046 IFQ$=CHR$(20)THENL=LEN(Z$):Z$=LEFT$
(Z$,L-1):PRINTCHR$(157);:GOTO8030
8050 Z$=Z$+Q$:PRINTQ$;:GOTO8030
8060 X=X+1
8070 PRINT#1,Z$:PRINT:GOTO8010
8080 W$=""
8090 PRINT"▒▒WOLLEN SIE NOCH MEHR INFOS
 EINGEBEN ?"
8100 RETURN
8110 :
8120 :
8130 :
9000 PRINT"▒▒▒▒▒▒"
9010 IFG=1THENPRINT"DATEI NICHT VORHANDE
N":FORA=1TO2000:NEXT:GOTO1000
9020 FORA=1TO12:FORB=1TO39:PRINTCHR$(46)
;:NEXT:PRINT:NEXT
9030 RETURN
10000 OPEN1,8,2,"AKTUELL,S,R":OPEN2,8,15
:INPUT#2,A:IFA=0THEN10020
10010 PRINT"_EHLER";A:CLOSE1:CLOSE2:RUN
10020 INPUT#1,A$:PRINTA$:IFST><66THEN100
20
10030 CLOSE1:CLOSE2:RUN
READY.
```

Listing 3 ist das Mailbox-Ladeprogramm. Es muß vor Listing 4 eingeladen und gestartet werden.

```
900 PRINTCHR$(14):PRINT"▒
950 P$="...................................
......
1000 PRINT"_ENN ♥IE EIN EIGENES  MPRESSU
M"
1010 PRINT"EINGEBEN WOLLEN DRUECKEN ♥IE
BITTE <I>.":PRINT
1020 PRINT"_ENN ♥IE SCHON EIN  MPRESSUM
HABEN "
1030 PRINT"DANN <H>
1040 GETQ$:IFQ$=""THEN1040
1050 IFQ$="I"THEN1110
1060 IFQ$="H"THENPRINT:PRINT"| ¬ | | ¬
C ♠ _ | ¬ ✓":GOTO2000
1070 GOTO1040
1080 :
1090 REM           EINGABE IMPRESSUM
1100 :
1110 PRINT"▒":PRINT
1120 PRINT"|ITTE GEBEN ♥IE ,HR ,MPRESSUM
 EIN
1130 PRINT"\AXIMAL 10 ♦EILEN A 35 ♦EICHE
N"
1140 PRINT
1150 OPEN1,3,0:OPEN2,0,1
1160 FORZ=1TO10:PRINTZ;TAB(4):INPUT#2,FE
$(Z):PRINT:NEXT:CLOSE2:CLOSE1
1270 PRINT"♠LLES ┌┘ ?";
1280 GETA$:IFA$=""THEN1280
1290 IFA$="J"ORA$="Y"THEN1320
1300 IFA$="N"THENPRINT"▒▒▒▒▒":GOTO1150
1310 GOTO1280
1320 OPEN8,8,5,"@:IMPRESSUM,U,W"
1330 FORA=1TO10:PRINT#8,FE$(A):NEXT
1340 CLOSE8
1350 PRINT"▒
1360 PRINT"|ITTE GEBEN ♥IE JETZT ,HR |AU
PTMENUE EIN":PRINT
1370 PRINT"|¬ ♥¬ ¬L":PRINT
1380 PRINT"1. ♠KTUELLE ,NFORMATIONEN"
1390 PRINT"2. ¬ATALOG"
1400 PRINT"3. ¬LECTRONIC \AIL"
1410 PRINT"▒▒▒▒▒▒":OPEN1,3,0:OPEN2,0,1
1420 FORA=1TO3:INPUT#2,FE$(A):PRINT:NEXT
:CLOSE2:CLOSE1
1460 FE$(1)=FE$(1)+RIGHT$(P$,36-LEN(FE$(
1)))+"<1>"
1470 FE$(2)=FE$(2)+RIGHT$(P$,36-LEN(FE$(
2)))+"<2>"
1480 FE$(3)=FE$(3)+RIGHT$(P$,36-LEN(FE$(
3)))+"<3>"
1490 PRINT:PRINT
1500 FORA=1TO3:PRINTFE$(A):NEXT:PRINT
1510 PRINT"¬INGABE ┌┘ ?"
1520 GETA$:IFA$=""THEN1520
1530 IFA$="J"ORA$="Y"THEN1560
1540 IFA$="N"THEN1350
1550 GOTO1520
1560 OPEN8,8,5,"IMPRESSUM,U,A"
1570 FORA=1TO3:PRINT#8,FE$(A):NEXT
1580 CLOSE8
2000 :
2010 REM      TIME OUT EINSTELLEN
2020 :
2030 PRINT:PRINT:PRINT
2040 PRINT" ITTE GEBEN ♥IE NUN DEN  IMEO
UT EIN
2050 PRINT" AS HEISST,WIE LANGE EIN ♠NRU
FER IN
2060 PRINT"DER  AILBOX ARBEITEN DARF"
2070 PRINT"     50  INUTEN
2080 INPUTT:IFT[illegible]THEN2080
2090 PRINTT"  INUTEN ┌┘"
2100 POKE2,T
10000 FORI=51200TO51481
10010 READX:POKEI,X:S=S+X:N=N+1:NEXT
10020 DATA173,14,220,9,128,141,14,220,17
3,15,220,41
10030 DATA127,141,15,220,32,121,,240,101
,32,253,174
10040 DATA32,158,173,32,163,182,201,6,20
8,107,160,
10050 DATA177,34,56,233,48,201,3,176,96,
10,10,10
10060 DATA10,133,251,200,177,34,56,233,4
8,201,10,176
10070 DATA80,5,251,208,4,169,146,208,15,
201,36,176
10080 DATA68,201,19,144,7,56,248,233,18,
216,9,128
10090 DATA141,11,220,32,253,200,141,10,2
20,32,253,200
10100 DATA141,9,220,169,,141,8,220,32,12
1,,240
10110 DATA13,32,253,174,32,158,183,224,1
6,176,22,142
10120 DATA167,2,120,173,20,3,73,161,141,
20,3,173
10130 DATA21,3,73,34,141,21,3,88,96,76,7
2,178
10140 DATA165,251,72,165,252,72,173,136,
2,133,252,169
10150 DATA,133,251,160,30,173,11,220,201
,18,240,17
10160 DATA201,128,144,15,41,127,201,18,2
40,9,248,24
10170 DATA105,18,216,208,2,169,,32,219,2
00,173,10
10180 DATA220,32,219,200,173,9,220,32,21
9,200,173,8
10190 DATA220,9,48,32,243,200,104,133,25
2,104,133,251
10200 DATA76,49,234,72,41,240,74,74,74,7
4,24,105
10210 DATA48,32,243,200,104,41,15,24,105
,48,32,243
10220 DATA200,169,58,145,251,173,167,2,1
53,,216,200
10230 DATA96,200,177,34,56,233,48,201,6,
176,134,10
10240 DATA10,10,10,133,251,200,177,34,56
,233,48,201
10250 DATA10,176,238,5,251,96
15000 OPEN1,8,2,"IMPRESSUM,U,R"
15010 INPUT#1,A$:PRINTA$:IFST=0THEN15010
15020 CLOSE1:PRINT:PRINT"OK J/N"
15030 WAIT198,1:GETA$:IFA$="N"THENRUN
20000 PRINT"▒▒BITTE LADEN ♥IE NUN MBA2 N
ACH!
READY.
```

Listing 4 ist das eigentliche Mailboxprogramm. Es wird nachgeladen sobald das Programm aus Listing 3 Sie dazu auffordert.

```
0 OPEN 1,2,2,CHR$(166)+CHR$(224)
1 POKE53281,0:POKE53280,43
25 OPEN9,8,5,"♠NRUFER,S,R":INPUT#9,AN%:C
LOSE9
30 POKE56579,PEEK(56579)OR8:POKE56577,0
1000 REM *******************************
1010 REM ***       M A I L B O X      ***
1020 REM ***    ALL COPYRIGHTS BY     ***
1030 REM ***      THOMAS  KOETHER     ***
1035 REM *** UND RUDOLF MOELLENBECK   ***
1040 REM *******************************
1050 :
1060 REM        INITIALISIERUNG
1070 :
1080 PRINT"▒▒":PRINT"\AILBOX ,NITIALISIE
RUNG...
1100 PRINTCHR$(14)
1110 BL$="
        "
1120 P$="...............................
....."
1130 DIMNA$(40):DIMPA$(40):DIMBR$(16):DI
MEM$(11):DIMBE$(100)
1140 DIMO%(256):DIMI%(256):GOSUB5700
1150 OPEN2,8,3,"NAMEN,S,R":INPUT#2,N$:AN
=VAL(N$)
1160 IFAN>0THENFORA=1TOAN:INPUT#2,N$:NA$
(A)=N$:NEXT
1170 CLOSE2
1180 E$=",HRE ¬INGABE...................
...? :"
1190 FORX=1TO5:READA$:ME$(X)=A$:NEXT:FOR
X=1TO4:READA$:IN$(X)=A$:NEXT
1200 OPEN2,8,5,"IMPRESSUM,U,R":FORA=1TO1
0:INPUT#2,IM$(A):NEXT
1210 FORA=1TO3:INPUT#2,ME$(A):NEXT:CLOSE
2
1220 FORA=1TO11:READA$:EM$(A)=A$:NEXT
1230 OPEN2,8,3,"PASSWORT,S,R"
1240 INPUT#2,N$:AN=VAL(N$)
1250 IFAN>0THENFORA=1TOAN:INPUT#2,P$:PA$
(A)=P$:NEXT
1260 CLOSE2
1270 T=PEEK(2):TT=T:OPEN15,8,15:INPUT#15
,A%:CLOSE15
1275 :
1280 PRINT"▒":PRINT"\AILBOX 64 ▒,CH WART
E AUF ♠NRUF":A=PEEK(56577)
1285 PRINT"♦UM MANUELLEN ♥TART |ASTE DRU
ECKEN"
1290 PRINTPEEK(56577):GETA$:IFA=PEEK(565
77)ANDA$=""THENPRINT"▯";:GOTO1290
1295 IFA$><""THEN1305
1300 AR=AR+1:IFAR<10THEN1290
1302 POKE56577,PEEK(56577)OR8
1305 AN%=AN%+1:OPEN9,8,5,"@:♠NRUFER,S,W"
:PRINT#9,AN%:CLOSE9
1310 FORA=1TO25:PRINT:PRINT#1:NEXT:PRINT
:PRINT#1:Z$=STR$(AN%)
1311 Z$=RIGHT$(Z$,LEN(Z$)-1):PRINT"
 ♥IE SIND DER ";Z$;". ♠NRUFER
1312 PRINT#1,"        ♥IE SIND DER ";Z$;"
. ♠NRUFER":FORA=1TO2000:NEXT
1315 SYS51200,"000000",1:TI$="000000"
1320 GOSUB 5800:REM   BEGRUESSUNGS BILD
1330 :
1340 REM       MODEM ABFRAGE
1350 :
1360 REM              MENUE
1370 :
1380 FORA=1TO5:PRINT#1:NEXT
1390 FORA=1TO5:PRINT:NEXT
1400 PRINT#1,"            |A U P T M E N
U E"
1410 PRINT"             |A U P T M E N U E
"
1420 PRINT#1,"           ------------------
----"
1430 PRINT"            --------------------
-"
1440 PRINT#1:PRINT#1
1450 PRINT:PRINT
1460 FORA=1TO5:IFA=4THEN1490
1470 PRINT#1,ME$(A):GOSUB1510
1480 PRINTME$(A)
1490 NEXT
1500 GOTO 1790
1510 :
1520 REM             ANTWORT
1530 :
1540 GOSUB9000:IFT%>TTHEN8030
1550 GET#1,Q$:IFQ$=""THEN1630
1560 Q=O%(ASC(Q$)):Q$=CHR$(I%(ASC(Q$)))
1570 IFQ=19THEN1720
1580 IFQ=24THENGOTO8030
1590 RETURN
1600 :
1610 REM          EIGENE EINGABE
1620 :
1630 GETR$:IFR$=""THENRETURN
1640 IFR$="^"THENPRINT:PRINT"< ";:GOTO 1
660
1642 IFR$=CHR$(20)THENR$=CHR$(8)
1645 Q$=R$
1650 RETURN
1660 GETR$:IFR$=""THEN1660
1670 IFR$=CHR$(13)THENPRINT#1,R$:PRINTR$
```

```
:RETURN
1680 PRINT#1,R$;:PRINTR$;:GOTO1660
1690 :
1700 REM      HALTESCHLEIFE
1710 :
1720 GET#1,Q$:IFQ$=""THEN1720
1730 Q=O%(ASC(Q$)):Q$=CHR$(I%(ASC(Q$)))
1740 IFQ=17THENRETURN
1750 IFQ=24THENGOTO8030
1760 PRINT#1,Q$
1770 PRINTQ$:GOTO1720
1780 :
1790 REM         EINGABE ONLINE
1800 :
1810 PRINT#1
1820 PRINT
1830 PRINT#1,E$;:GOSUB1510
1840 PRINTE$;
1850 GOSUB1510
1860 IFQ$=""THEN1850
1870 Q=O%(ASC(Q$)):Q$=CHR$(I%(ASC(Q$))):
PRINT#1,Q$:PRINTQ$
1880 IFQ=49THEN1960
1890 IFQ=50THEN2230
1900 IFQ=51THEN3120
1910 IFQ=52THEN1850
1920 IFQ=48THENPRINT#1,"AUF WIEDERSEHEN"
:PRINT"AUF WIEDERSEHEN":GOTO8030
1930 IFQ=24THEN8030
1940 GOTO1850
1950 :
1960 REM     AKTUELLE INFORMATIONEN
1970 :
1980 FORA=1TO5:PRINT#1:PRINT:NEXT:GOSUB1
510
1990 PRINT#1,"                / E W S"
2000 PRINT"              / E W S"
2010 PRINT#1:PRINT#1
2020 PRINT:PRINT
2030 OPEN2,8,3,"AKTUELL,S,R":W$=""
2040 INPUT#2,W$:W$="":IFST=64THEN2090
2050 GET#2,H$:IFH$<>CHR$(13)THENW$=W$+H$
:GOTO2050
2060 IFST=64THEN2090
2070 IFW$="♠♠♠"THEN2160
2080 GOTO2150
2090 PRINT#1:PRINT#1:PRINT#1,"- N D E  D
E R  ⌐ N F O S":W$=""
2100 PRINT:PRINT:PRINT"- N D E  D E R  ⌐
 N F O S":W$=""
2110 PRINT#1:PRINT#1,"| A S T E  D R U E
 C K E N"
2120 PRINT:PRINT"| A S T E  D R U E C K
E N"
2130 GOSUB1510:IFQ$=""THEN2130
2140 FORA=1TO5:PRINT#1:PRINT:NEXT:GOSUB1
510:CLOSE2:GOTO1360
2150 PRINT#1,W$:PRINTW$:W$="":GOSUB1510:
GOTO2050
2160 W$="":PRINT#1:PRINT#1,"     OEITER (
_ETURN) SONST (0)
2170 W$="":PRINT:PRINT"     OEITER (_ETUR
N) SONST (0)
2180 GOSUB1510
2190 IFQ=13THENPRINT#1,Q$:PRINTQ$:GOTO20
50
2200 IFQ=48THENCLOSE2:GOTO1360
2210 GOTO 2180
2220 :
2230 :
2240 REM       CATALOG
2250 :
2260 :
2270 FORA=1TO5:PRINT#1:PRINT:NEXT:GOSUB1
540
2280 OPEN3,8,3,"CATALOG,S,R":C$="":AN=0
2290 INPUT#3,C$:C$="":IFST=64THEN3050
2300 Z=1:S=9:FORA=1TO5:PRINT#1:PRINT:NEX
T
2310 GET#3,H$:IFH$<>CHR$(13)THENC$=C$+H$
:GOTO2310
2320 IFST=64THENFL=8:GOTO2390
2330 IFC$="♠♠♠"THEN2390
2335 Z$=STR$(Z):Z$=RIGHT$(Z$,LEN(Z$)-1)
2340 PRINT#1,Z$"."C$:PRINTZ$"."C$:GOSUB1
540
2350 FE$(Z)=C$:Z=Z+1:S=S-1:C$="":GOTO231
0
2360 :
2370 REM       CATALOG MENUE
2380 :
2390 C$="":PRINT#1:PRINT#1,"\OECHTEN ♥IE
 ETWAS BESTELLEN  <B>"
2400 PRINT:PRINT"\OECHTEN ♥IE ETWAS BEST
ELLEN  <B>"
2410 IFFL=8THEN2440
2420 PRINT#1,"DIE _ISTE WEITER SEHEN
  <W>"
2430 PRINT"DIE _ISTE WEITER SEHEN
  <W>"
2440 PRINT#1,"ODER ZURUECK ZUM \ENUE
  <M>";" "
2450 PRINT"ODER ZURUECK ZUM \ENUE
  <M>";" "
2460 GOSUB1540
2470 IFQ$=""THEN2460
2480 IFQ=98ORR$="B"THEN2550
2490 IFQ=119ORR$="W"THENZ=1:FORA=1TO5:PR
INT#1:PRINT:NEXT:GOTO2300
2500 IFQ=109ORR$="M"THENCLOSE3:GOTO2910
2510 GOTO2460
2520 :
2530 REM        BESTELLEN
2540 :
2550 IFOP=1THEN2720
2560 OP=1:PRINT#1:PRINT
2570 PRINT#1,"|ITTE GEBEN ♥IE ⌐HREN /AME
N EIN":N$=""
2580 PRINT"|ITTE GEBEN ♥IE ⌐HREN /AMEN E
IN"
2590 GOSUB1540
2600 IFQ$=""THEN2590
2610 IFQ=13THEN2650
2620 :
2630 IFQ=20THENB$=N$:GOSUB6050:N$=B$:GOT
O2590
2640 N$=N$+Q$:PRINT#1,Q$;:PRINTQ$;:GOTO2
590
2650 PRINT#1:PRINT#1,"UND NUN ⌐HRE ♠DRES
SE"
2660 PRINT:PRINT"UND NUN ⌐HRE ♠DRESSE"
2670 GOSUB1540
2680 IFQ$=""THEN2670
2690 IFQ=13THEN2720
2700 IFQ=20THENB$=AD$:GOSUB6050:AD$=B$:G
OTO2670
2710 AD$=AD$+Q$:PRINT#1,Q$;:PRINTQ$;:GOT
O2670
2720 PRINT#1:PRINT
2730 PRINT#1,"|ITTE GEBEN ♥IE DIE |ITELN
UMMER EIN"
2740 PRINT"|ITTE GEBEN ♥IE DIE |ITELNUMM
ER EIN":AN=AN+1
2750 GOSUB1540
2760 IFQ$=""THEN2750
2770 IFQ<49ORQ>57-STHEN2750
2780 PRINT#1,Q$:PRINTQ$:T=VAL(Q$):PRINT#
1:PRINT:PRINT#1,FE$(T):PRINTFE$(T)
2790 BE$(AN)=FE$(T)
2800 PRINT#1,"OOLLEN ♥IE NOCH ETWAS BEST
ELLEN ?";
2810 PRINT"OOLLEN ♥IE NOCH ETWAS BESTELL
EN ?";
2820 GOSUB1540
2830 IFQ$=""THEN2820
2840 IFQ=106ORQ=121THENPRINT#1,Q$:PRINTQ
$:GOTO2720
2850 IFQ=110ANDFL<>8THENPRINT#1,Q$:PRINT
Q$:GOTO2300
2860 IFFL=8THEN3050
2870 GOTO2820
2880 :
2890 REM        BESTELLISTE ABSAVEN
2900 :
2910 PRINT#1:PRINT
2920 PRINT#1,"-IE BESTELLTEN ♠RTIKEL WER
DEN IN EIN"
2930 PRINT"-IE BESTELLTEN ♠RTIKEL WERDEN
 IN EIN"
2940 PRINT#1,"PAAR |AGEN BEI ⌐HNEN SEIN"
2950 PRINT"PAAR |AGEN BEI ⌐HNEN SEIN"
2960 OPEN10,8,10,"BESTELLISTE,S,A"
2970 IFAN=0THENCLOSE10:GOTO1370
2980 FORA=1TOAN:PRINT#10,BE$(A):NEXT
2990 PRINT#10,"♠♠♠":CLOSE10
3000 GOSUB1540
3010 GOTO1370
3020 :
3030 REM          CATALOG ZUENDE
3040 :
3050 PRINT#1:PRINT#1:PRINT#1,"- ♠ | ♠ L
Γ |  ♦ ⌐ - Γ / - -":PRINT#1:PRINT#1
3060 PRINT:PRINT:PRINT"- ♠ | ♠ L Γ |  ♦
 ⌐ - / - -":PRINT:PRINT
3070 PRINT#1,"| A S T E  D R U E C K E N
"
3080 PRINT"| A S T E  D R U E C K E N"
3090 GOSUB1540
3100 IFQ$=""THEN3090
3110 CLOSE3:GOTO1370
3120 :
3130 REM    ELECTRONIC MAIL ENTRY
3140 :
3150 FORA=1TO5:PRINT#1:PRINT:NEXT
3160 PRINT#1,"       - L - - | _ Γ / ⌐ -
 \ ♠ ⌐ L"
3170 PRINT"       - L - - | _ Γ / ⌐ - \
♠ ⌐ L"
3180 PRINT#1,"       --------------------
----------"
3190 PRINT"      -------------------------
------"
3200 PRINT#1:PRINT#1
3210 PRINT:PRINT
3220 FORA=1TO5
3230 PRINT#1,EM$(A):GOSUB1510
3240 PRINTEM$(A)
3250 NEXT:PRINT#1:PRINT
3260 PRINT#1,E$;:PRINTE$
3270 GOSUB1510
3280 IFQ$=""THEN3270
3290 IFQ<48ORQ>52THEN3270
3300 PRINT#1,Q$:PRINTQ$
3310 IFQ=48THEN1360
3320 IFQ=49THEN3390
3330 IFQ=50THEN4020
3340 IFQ=51THEN4610
3350 GOTO5160 :REM    Q=52
3360 :
3370 REM        MENUEPUNKT 1
3380 :
3390 FORA=1TO5:PRINT#1:PRINT:NEXT:GOSUB1
510
3400 PRINT#1,EM$(6):PRINT#1,EM$(7):N$=""
3410 PRINTEM$(6):PRINTEM$(7):N$=""
3420 GOSUB1510
3430 IFQ$=""THEN3420
3440 IFQ=13THENFL=1:PRINT#1,Q$:PRINTQ$:G
OTO3520
3450 IFQ=20THENB$=N$:GOSUB6050:N$=B$:GOT
O3420
3460 PRINT#1,Q$;
3470 PRINTQ$;
3480 N$=N$+Q$:GOTO3420
3490 :
3500 REM     NAMENS KONTROLLE
3510 :
3520 IFLEN(N$)=0THEN3120
3530 Z=1
3540 IFNA$(Z)=N$THENONFLGOTO3620,4140,52
70
3550 IFZ<20THENZ=Z+1:GOTO3540
3560 PRINT#1:PRINT
3570 PRINT#1,N$;" HAT KEIN ¬OSTFACH BEI
UNS !!!"
3580 PRINTN$;" HAT KEIN ¬OSTFACH BEI UNS
 !!!":FORA=1TO2000:NEXT:GOTO3120
3590 :
3600 REM          BRIEF EINGABE
3610 :
3620 OPEN15,8,15,"IO":PRINT#15,"M-R" CHR
$(250) CHR$(2)
3630 GET#15,L$:IFL$=""THENL$=CHR$(0)
3640 PRINT#15,"M-R" CHR$(252) CHR$(2):GE
T#15,M$:IFM$=""THENM$=CHR$(0)
3650 FR=ASC(L$)+256*ASC(M$):CLOSE15
3660 IFFR>15THEN3720
3670 PRINT#1:PRINT#1,"_EIDER KEIN ♥PEICH
ERPLATZ MEHR           VORHANDEN !!!"
3680 PRINT:PRINT"_EIDER KEIN ♥PEICHERPLA
TZ MEHR          VORHANDEN !!!"
3690 FORA=1TO2000:NEXT:GOTO3120
3700 :
3710 :
3720 OPEN2,8,3,N$+PA$(Z)+",S,A"
3730 PRINT#1,"|EBEN ♥IE ⌐HRE /ACHRICHT E
IN           MAX. 15 ♠EILEN"
3740 PRINT"|EBEN ♥IE ⌐HRE /ACHRICHT EIN
         MAX. 15 ♠EILEN"
3750 PRINT#1:PRINT#1,"♠BSCHLUSS MIT _EER
ZEILE":PRINT#1
3760 PRINT:PRINT"♠BSCHLUSS MIT _EERZEILE
":PRINT
3770 Z=1:W$=" "
3780 IFZ=16THENPRINT#1:PRINT#1,"♠EILENGR
ENZE ERREICHT"
3790 IFZ=16THENPRINT:PRINT"♠EILENGRENZE
ERREICHT":GOTO3950
3800 GET#1,Q$:IFQ$=""THEN3800
3810 Q$=CHR$(I%(ASC(Q$)))
3820 IFASC(Q$)=13ANDASC(W$)=13THENBR$(Z)
="♠♠♠":GOTO3950
3830 IFASC(Q$)=13THENW$=Q$:PRINT#1:PRINT
:BR$(Z)=B$:B$="":Z=Z+1:GOTO3780
3840 IFASC(Q$)<>20THEN3870
3850 GOSUB6050
3860 GOTO3800
3870 PRINT#1,Q$;:PRINTQ$;
3880 B$=B$+Q$:W$=Q$
3890 IFLEN(B$)<80THEN3800
3900 PRINT#1,"♠ ¬ ⌐ L ¬  ♠  L ♠ / ! !!
!":B$="":PRINT#1
3910 PRINT"♠ ¬ ⌐ L ¬  ♠ ⌐  L ♠ / ! !!!":
B$="":PRINT:GOTO3800
3920 :
3930 REM        BRIEF ABSAVEN
3940 :
3950 PRINT#1:PRINT#1,"|RIEF WIRD ABGESPE
ICHERT"
3960 PRINT:PRINT"|RIEF WIRD ABGESPEICHER
T"
3970 FORA=1TOZ:PRINT#2,BR$(A):NEXT:CLOSE
2:GOTO3120
3980 :
3990 :
4000 REM       MENUEPUNKT 2
4010 :
4020 FORA=1TO5:PRINT#1:PRINT:NEXT:GOSUB1
510
4030 PRINT#1,EM$(8):N$=""
4040 PRINTEM$(8):N$=""
4050 GOSUB1510
4060 IFQ$=""THEN4050
4070 IFQ=13THENFL=2:PRINT#1,Q$:PRINTQ$:G
OTO3520
4080 IFQ=20THENB$=N$:GOSUB6050:N$=B$:GOT
O4050
4090 PRINT#1,Q$;:PRINTQ$;
4100 N$=N$+Q$:GOTO4050
4110 :
4120 REM        PASSWORT EINGABE
4130 :
4140 FL=1
4150 PRINT#1,EM$(10):P$=""
4160 PRINTEM$(10):P$=""
4170 GOSUB1510
4180 IFQ$=""THEN4170
4190 IFQ=13THENPRINT#1,Q$:PRINTQ$:GOTO42
60
4200 IFQ=20THENB$=P$:GOSUB6050:P$=B$:GOT
O4170
4210 PRINT#1,Q$;:PRINTQ$;
4220 P$=P$+Q$:GOTO4170
4230 :
4240 REM         PASSWORT KONTROLLE
4250 :
4260 IFLEN(P$)=0THEN3120
4270 IFPA$(Z)=P$THENONFLGOTO4330,5310
4280 PRINT#1:PRINT#1,"_ALSCHE ¬INGABE !!
!"
4290 PRINT:PRINT"_ALSCHE ¬INGABE !!!":GO
```

Fortsetzung Listing 4

```
TO4150
4300 :
4310 REM           BRIEF AUSGABE
4320 :
4330 OPEN2,8,3,NA$(Z)+PA$(Z)+",S,R":T$="
"
4340 FORA=1TO5:PRINT#1:PRINT:NEXT
4350 INPUT#2,T$:IFST=64THEN4510
4360 T$=""
4370 GET#2,H$:IFH$<>CHR$(13)THENT$=T$+H$
:GOTO4370
4380 IFST=64THEN4510
4390 IFT$="♠♠♠"THEN4410
4400 PRINT#1,T$:PRINTT$:T$="": GOSUB1510
:GOTO4370
4410 PRINT#1:PRINT#1,"○EITER (_ETURN) ¬N
DE (0)":T$=""
4420 PRINT:PRINT"○EITER (_ETURN) ¬NDE (0
)":T$=""
4430 GOSUB1510
4440 IFQ$=""THEN4430
4450 IFQ=13THENFORA=1TO5:PRINT#1:PRINT:N
EXT:GOTO4370
4460 IFQ=48THENCLOSE2:GOTO3120
4470 GOTO4430
4480 :
4490 REM           BRIEFE ZUENDE
4500 :
4510 PRINT#1:PRINT#1,"/EINE WEITEREN |RI
EFE VORHANDEN !!!"
4520 CLOSE2:PRINT:PRINT"/EINE WEITEREN |
RIEFE VORHANDEN !!!"
4530 PRINT#1:PRINT#1,"| ASTE DRUECKEN"
4540 PRINT:PRINT"| ASTE DRUECKEN"
4550 GOSUB1510
4560 IFQ$=""THEN4550
4570 GOTO3120
4580 :
4590 REM         MAILBOX EINRICHTEN
4600 :
4610 IFAN=40THENPRINT#1,EM$(11):PRINTEM$
(11):FORN=1TO3000:NEXT:GOTO3120
4620 FORA=1TO5:PRINT#1:PRINT:NEXT:GOSUB1
510
4630 PRINT#1,EM$(8):N$="":Z=1
4640 PRINTEM$(8):N$=""
4650 GOSUB1510
4660 IFQ$=""THEN4650
4670 IFZ<1THENZ=1
4680 IFQ=13THENPRINT#1,Q$:PRINTQ$:GOTO47
20
4690 IFQ=20THENZ=Z-1:B$=N$:GOSUB6050:N$=
B$:GOTO4650
4700 IFZ<9THENPRINT#1,Q$;:PRINTQ$;:N$=N$
+Q$:Z=Z+1
4710 GOTO4650
4720 IFLEN(N$)=0THEN3120
4730 Z=1
4740 IFNA$(Z)=N$THEN4770
4750 IFZ<19THENZ=Z+1:GOTO4740
4760 GOTO4820
4770 PRINT#1:PRINT#1,"/AME SCHON EXISTEN
T":PRINT
4780 PRINT:PRINT"/AME SCHON EXISTENT":PR
INT:GOTO4630
4790 :
4800 REM        PASSWORT DEFINIEREN
4810 :
4820 PRINT#1,EM$(9):P$="":Z=1
4830 PRINTEM$(9):P$=""
4840 GOSUB1510
4850 IFQ$=""THEN4840
4860 IFQ=13THEN4910
4870 IFZ<1THENZ=1
4880 IFQ=20THENZ=Z-1:B$=P$:GOSUB6050:P$=
B$:GOTO4840
4890 IFZ<9THENPRINT#1,Q$;:PRINTQ$;:P$=P$
+Q$:Z=Z+1
4900 GOTO4840
4910 IFLEN(P$)=0THEN3120
4920 Z=1
4930 IFPA$(Z)=P$THEN4960
4940 IFZ<AN-1THENZ=Z+1:GOTO4930
4950 GOTO5020
4960 PRINT#1:PRINT#1,"¬ASSWORD SCHON EXI
STENT"
4970 PRINT:PRINT"¬ASSWORD SCHON EXISTENT
"
4980 GOTO4820
4990 :
5000 REM        NEUE MAILBOX CREIEREN
5010 :
5020 AN=AN+1:NA$(AN)=N$:PA$(AN)=P$
5030 PRINT#1:PRINT#1,"...| I T T E  W A
R T E N..."
5040 PRINT:PRINT"...| I T T E  W A R T E
 N...":GOSUB1510
5050 OPEN5,8,5,NA$(AN)+PA$(AN)+",S,W":PR
INT#5,"FIRST":CLOSE5
5060 OPEN5,8,5,"@:NAMEN,S,W":PRINT#5,AN:
FORA=1TOAN:PRINT#5,NA$(A):NEXT:CLOSE5
5070 OPEN5,8,5,"@:PASSWORT,S,W":PRINT#5,
AN:FORA=1TOAN:PRINT#5,PA$(A):NEXT
5080 CLOSE5
5090 PRINT#1:PRINT
5100 PRINT#1,"¬OSTFACH MIT DEM /AMEN ";N
$;"          EROEFFNET"
5110 PRINT"¬OSTFACH MIT DEM /AMEN ";N$;"
          EROEFFNET"
5120 GOTO3120
5130 :
5140 REM          MAILBOX LOESCHEN
5150 :
5160 FORA=1TO5:PRINT#1:PRINT:NEXT:GOSUB1
510
5170 PRINT#1,EM$(8):N$="":FL=3
5180 PRINTEM$(8):N$="":FL=3
5190 GOSUB1510
5200 IFQ$=""THEN5190
5210 IFQ=13THENFL=3:PRINT#1,Q$:PRINTQ$;:
GOTO3520
5220 IFQ=20THENB$=N$:GOSUB6050:N$=B$:GOT
O5190
5230 PRINT#1,Q$;:PRINTQ$;:N$=N$+Q$:GOTO5
190
5240 :
5250 REM             PASSWORT
5260 :
5270 FL=2:GOTO4150
5280 :
5290 REM          MAILBOX LOESCHEN
5300 :
5310 FORA=ZTOAN-1:NA$(A)=NA$(A+1):PA$(A)
=PA$(A+1):NEXT:NA$(AN)="":PA$(AN)=""
5320 AN=AN-1
5330 PRINT#1,"...| I T T E  W A R T E N.
.."
5340 PRINT"...| I T T E  W A R T E N..."
5350 OPEN5,8,5,"@:NAMEN,S,W"
5360 PRINT#5,AN:FORA=1TOAN:PRINT#5,NA$(A
):NEXT:CLOSE5
5370 OPEN5,8,5,"@:PASSWORT,S,W"
5380 PRINT#5,AN:FORA=1TOAN:PRINT#5,PA$(A
):NEXT:CLOSE5
5390 OPEN5,8,15,"S:"+N$+P$:CLOSE5
5400 PRINT#1:PRINT
5410 PRINT#1,"¬OSTFACH IST GELOESCHT"
5420 PRINT"¬OSTFACH IST GELOESCHT"
5430 GOTO3120
5440 :
5450 REM         MENUE - DATAS
5460 :
5470 DATA"♠KTUELLE ,NFORMATIONEN........
.....<1>"
5480 DATA"-ATALOG.......................
.....<2>"
5490 DATA"-LECTRONIC \AIL...............
.....<3>"
5500 DATA",NFO - ♥ERVICE................
.....<4>"
5510 DATA"¬NDE DER /OMMUNIKATION........
......<0>"
5520 :
5530 REM         INFO - DATAS
5540 :
5550 DATA"VC 64","VC 20","SPECTRUM","ATA
RI
5560 :
5570 REM    ELECTRONIC MAIL MENUE
5580 :
5590 DATA"|RIEF -INGABE.................
.....<1>"
5600 DATA"|RIEFE LESEN..................
.....<2>"
5610 DATA"¬OSTFACH EINRICHTEN...........
.....<3>"
5620 DATA"¬OSTFACH LOESCHEN.............
.....<4>"
5630 DATA"♠UM |AUPTMENUE................
.....<0>"
5640 DATA"^N WEN WOLLEN ♥IE EINEN |RIEF
5650 DATA"SCHREIBEN ?
5660 DATA"|ITTE GEBEN ♥IE ,HREN /AMEN EI
N          MAX. 8 ♠EICHEN   :"
5670 DATA"|ITTE DEFINIEREN ♥IE EIN ¬ASSW
ORT        MAX. 8 ♠EICHEN   :"
5680 DATA"|ITTE GEBEN ♥IE ,HR ¬ASSWORT E
IN :"
5690 DATA"\AXIMALE |EILNEHMERZAHL ERREIC
HT !!"
5700 REM                  ASCIICHANGE
5710 FORZ=32TO64:O%(Z)=Z:NEXT:O%(13)=13:
O%(20)=8:O%(160)=32
5720 FORZ=65TO90:Y=Z+32:O%(Z)=Y:NEXT:FOR
Z=91TO95:O%(Z)=Z:NEXT
5730 FORZ=193TO218:Y=Z-128:O%(Z)=Y:NEXT
5740 FORZ=133TO140:O%(Z)=Z:NEXT:O%(20)=2
0
5750 FORZ=0TO255:Y=O%(Z):IFY<>0THENI%(Y)
=Z
5760 NEXT:O%(17)=17:O%(19)=19:O%(24)=24
5770 RETURN
5780 :
5790 REM           EINGANGS BILD
5800 :
5820 PRINT#1,"++++++++++++++++++++++++++
+++++++++++++"
5830 PRINT"++++++++++++++++++++++++++++
++++++++++"
5840 PRINT#1,"+++      ♠ - L I Γ ♠ 6
 4        +++"
5850 PRINT"+++          \ ♠ - L I Γ ♠ 64
    +++"
5860 PRINT#1,"+++   COPYRIGHT ♥Γ_| ○♠_¬ ¬
 +++"
5870 PRINT"+++   COPYRIGHT ♥Γ_| ○♠_¬ ¬♠Γ_
 +++"
5880 PRINT#1,"++++++++++++++++++++++++++
+++++++++++++"
5890 PRINT"++++++++++++++++++++++++++++
++++++++++"
5900 FORA=1TO8:PRINT#1:PRINT:NEXT
5910 PRINT#1,"     -IESES \AILBOX ¬ROGRAM
M WURDE AUF"
5920 PRINT"    -IESES \AILBOX ¬ROGRAMM W
URDE AUF"
5930 PRINT#1,"    EINEN -OMMODORE X- 64
REALISIERT"
5940 PRINT"    EINEN -OMMODORE X- 64 REA
LISIERT"
5950 PRINT#1,"      GESCHRIEBEN VON |HOMA
S /OETHER"
5960 PRINT"     GESCHRIEBEN VON |HOMAS /
OETHER"
5970 PRINT#1,"         ADAPTIERT AUF ♠UTO
MODEM":PRINT#1:PRINT#1
5975 PRINT"         ADAPTIERT AUF ♠UTOMO
DEM
5976 PRINT#1,"            VON _UDOLF \OELLE
NBECK":PRINT#1:PRINT#1
5977 PRINT"           VON _UDOLF \OELLENBE
CK":PRINT:PRINT
5980 PRINT#1,"          | A S T E  D R U E C
 K E N"
5990 PRINT"          | A S T E  D R U E C K
E N"
6000 GET#1,Q$:IFQ$<>""THEN6010
6005 GETQ$:IFQ$=""THEN6000
6010 GOTO6070
6020 :
6030 REM            INST/DEL
6040 :
6050 L=LEN(B$):IFL>0THENB$=LEFT$(B$,L-1)
:PRINT#1,CHR$(20);:PRINTQ$;
6060 RETURN
6070 :
6080 REM            IMPRESSUM
6090 :
6100 FORA=1TO5:PRINT#1:PRINT:NEXT
6170 FORA=1TO10:PRINT#1,IM$(A):PRINTIM$(
A):NEXT
6180 FORA=1TO7:PRINT#1:PRINT:NEXT
6190 PRINT#1,"          | A S T E  D R U E C
 K E N"
6200 PRINT"          | A S T E  D R U E C K
E N"
6210 GET#1,Q$:IFQ$<>""THEN6220
6215 GETQ$:IFQ$=""THEN6210
6220 GOTO1380
8000 :
8020 :
8030 :
8040 SYS51200:PRINT#1:PRINT#1,"♠UF WIEDE
R SEHEN..."
8045 CLOSE2:CLOSE3:CLOSE4:CLOSE5:CLOSE6:
CLOSE7:CLOSE8:CLOSE9:CLOSE10
8050 POKE56577,0:GOTO1275
9000 T%=VAL(MID$(TI$,3,2)):IFT<T%THENRET
URN
9010 IFT-T%<TTTHENTT=TT-1
9020 RETURN
READY.
```

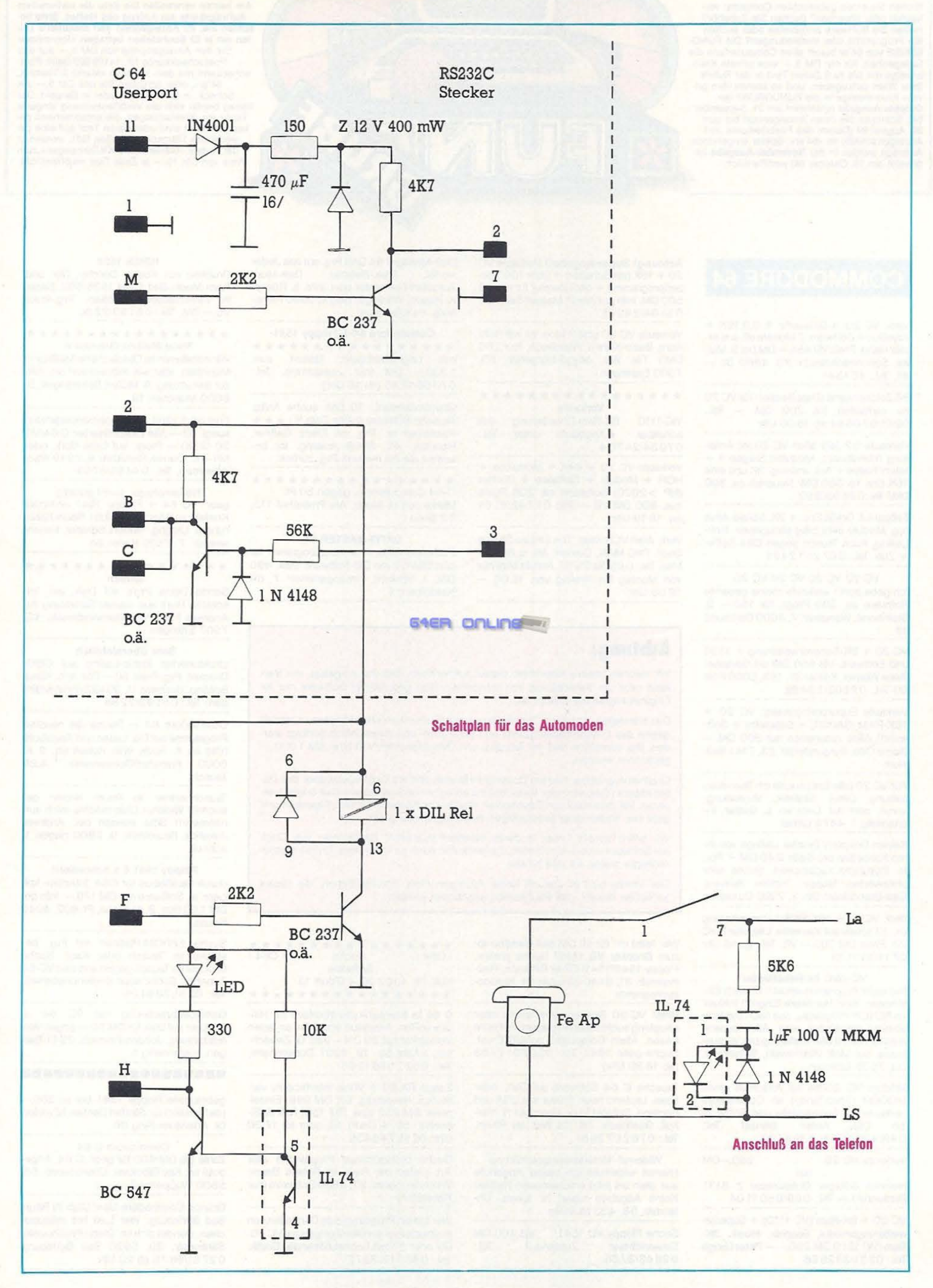
Steckbrief: Wer kennt Computer-Seminare?

Wer lernt schon gern allein? In der Gruppe ist der »Schüler« doch viel besser aufgehoben.

»Kurse und Seminare rund um den Computer« heißt das Thema in einer unserer nächsten Ausgaben. Wir möchten Sie informieren, wo welche Kurse zu welchem Preis angeboten werden, und wir möchten berichten, wie es Kursteilnehmern ergeht, beziehungsweise ergangen ist.

Dazu brauchen wir Sie. Schreiben Sie uns, welche Computer-Seminare Sie anbieten oder welche Sie besucht haben. Anbieter und Teilnehmer — wir sammeln Erfahrungen und Informationen von beiden Seiten. Übrigens, nicht nur »Programmieren in Basic, Pascal« und so weiter ist interessant, sondern alles, was man rund um den Computer mit anderen zusammen lernen kann.

Ihre Zuschriften senden Sie bitte an:
Redaktion
Computer persönlich
Karin Gößlinghoff
Hans-Pinsel-Str. 2
8013 Haar bei München



Schaltplan für das Automoden

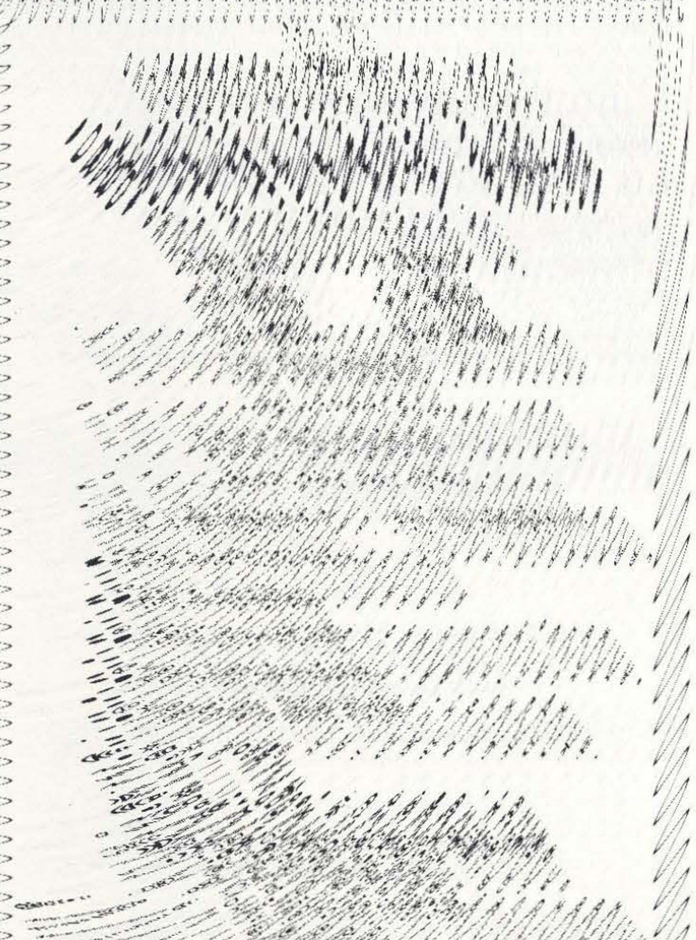
Anschluß an das Telefon

Wollen Sie einen gebrauchten Computer verkaufen oder erwerben? Suchen Sie Zubehör? Haben Sie Software anzubieten oder suchen Sie Programme oder Verbindungen? Die FUNDGRUBE von 64'er bietet allen Computerfans die Gelegenheit, für nur DM 5,— eine private Kleinanzeige mit bis zu 5 Zeilen Text in der Rubrik Ihrer Wahl aufzugeben. Und so kommt Ihre private Kleinanzeige in die FUNDGRUBE der **Oktober-Ausgabe** (erscheint am 21. September 84): Schicken Sie Ihren Anzeigentext bis zum 29. August 84 (Datum des Poststempels und Anzeigenschluß) an »64'er«. Später eingehende Aufträge werden in der **November-Ausgabe** (erscheint am 19. Oktober 84) veröffentlicht.



Am besten verwenden Sie dazu die vorbereitete Auftragskarte am Anfang des Heftes. Bitte beachten Sie: Ihr Anzeigentext darf maximal 5 Zeilen mit je 32 Buchstaben betragen. Überweisen Sie den Anzeigenpreis von DM 5,— auf das Postscheckkonto Nr. 14199-803 beim Postscheckamt mit dem Vermerk »Markt & Technik, 64'er«, oder schicken Sie uns DM 5,— als Scheck, in Briefmarken oder in Bargeld. Der Verlag behält sich die Veröffentlichung längerer Texte vor. Kleinanzeigen, die entsprechend gekennzeichnet sind oder deren Text auf eine gewerbliche Tätigkeit schließen läßt, werden in der Rubrik »Gewerbliche Kleinanzeigen« zum Preis von DM 10,— je Zeile Text veröffentlicht.



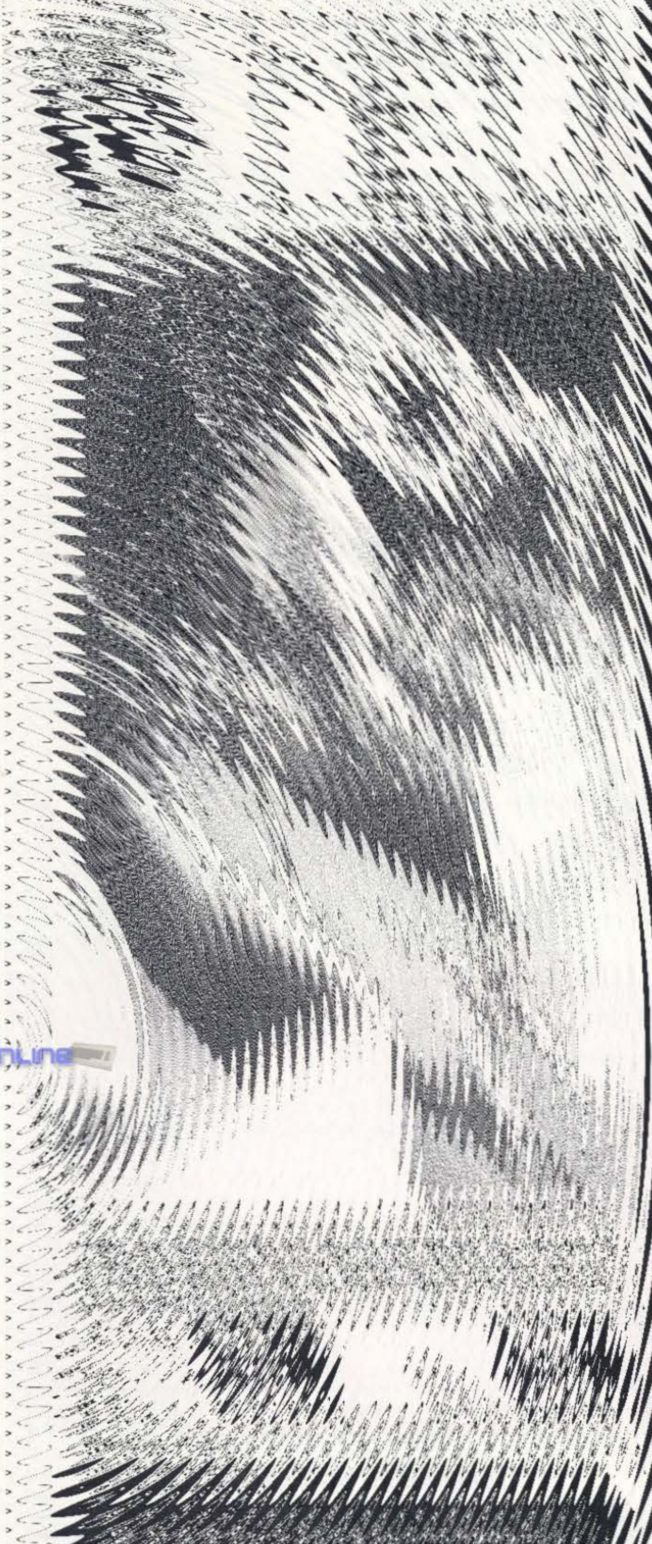




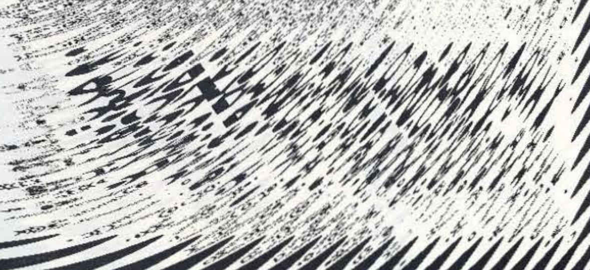


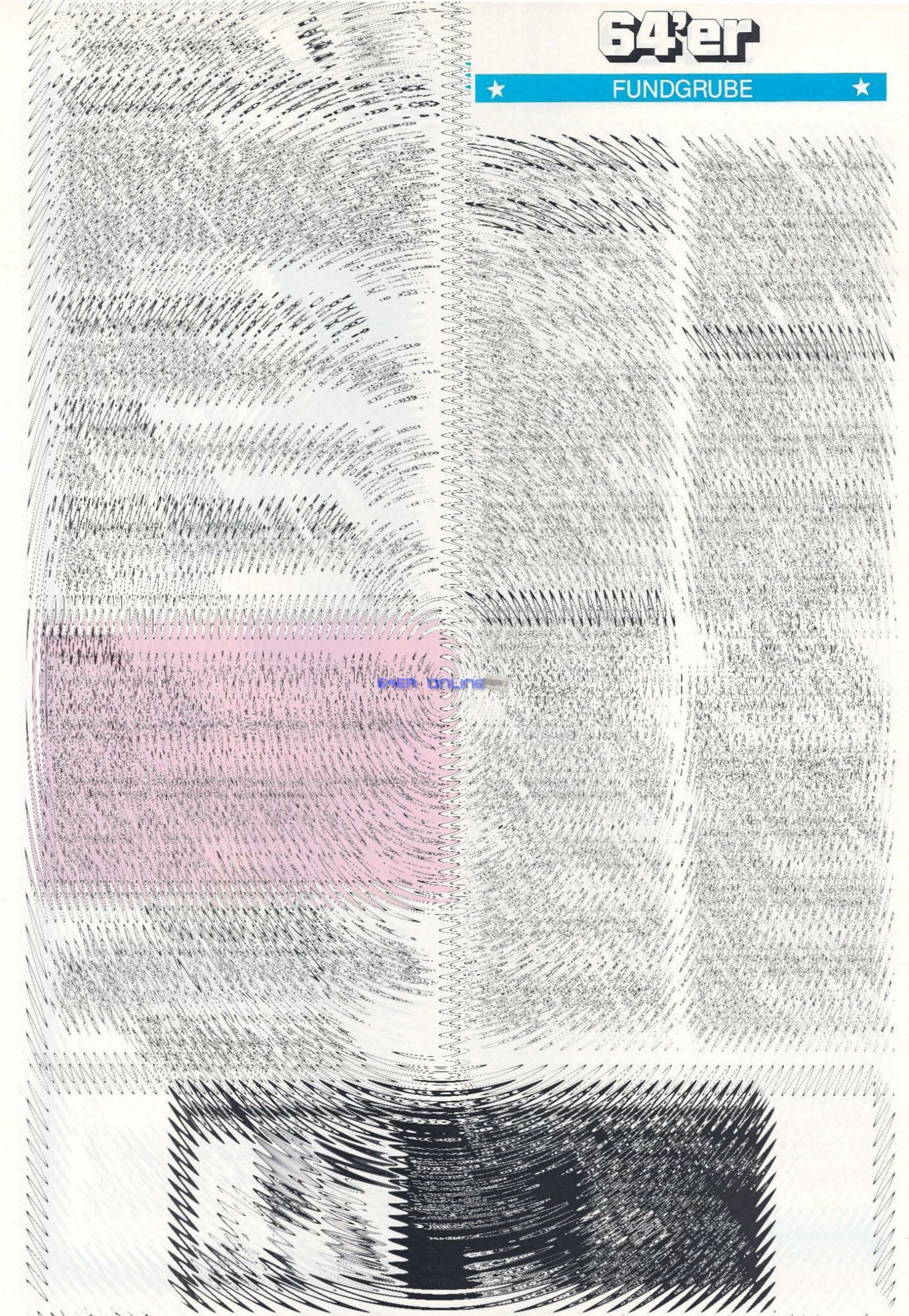


★ FUNDGRUBE ★





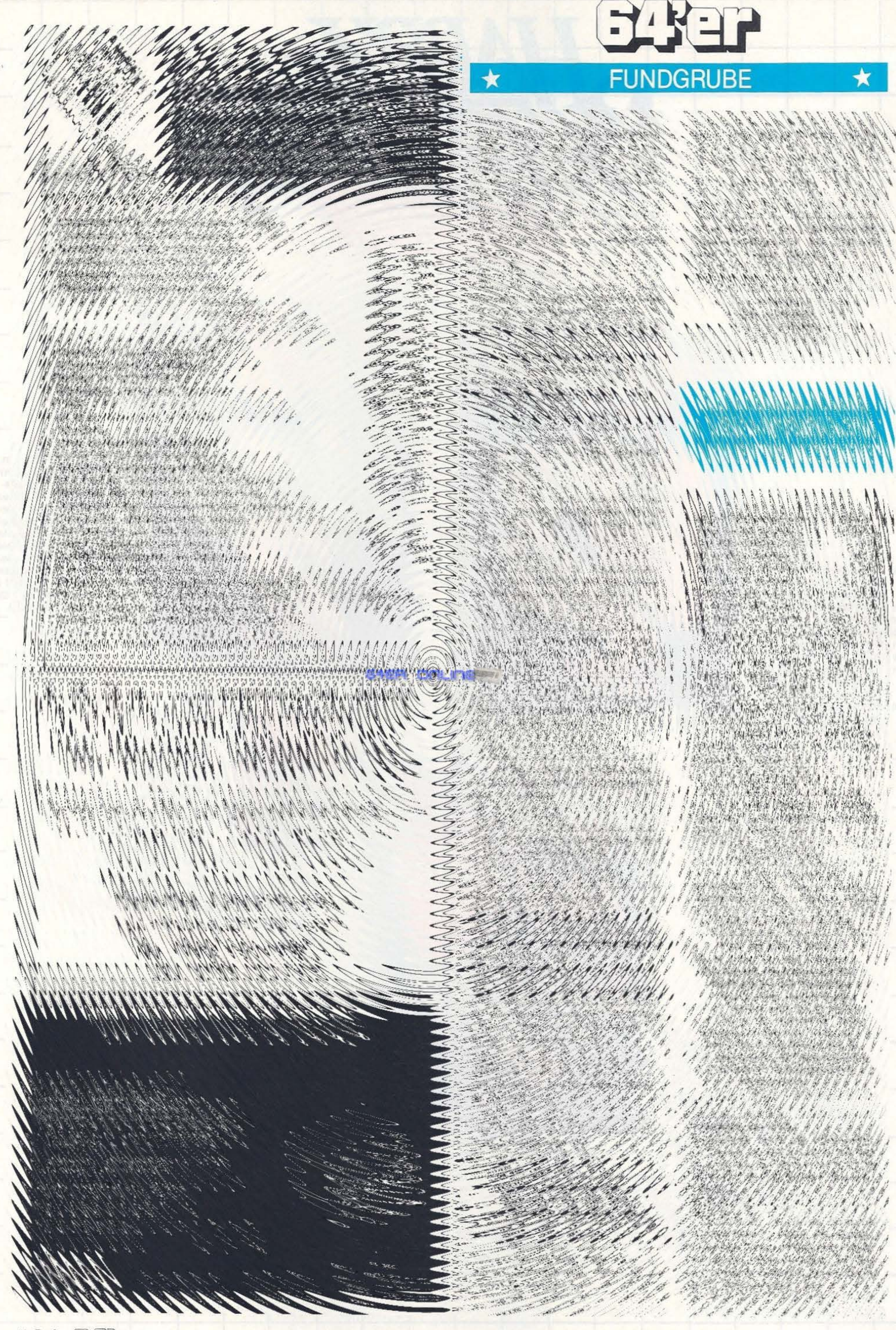






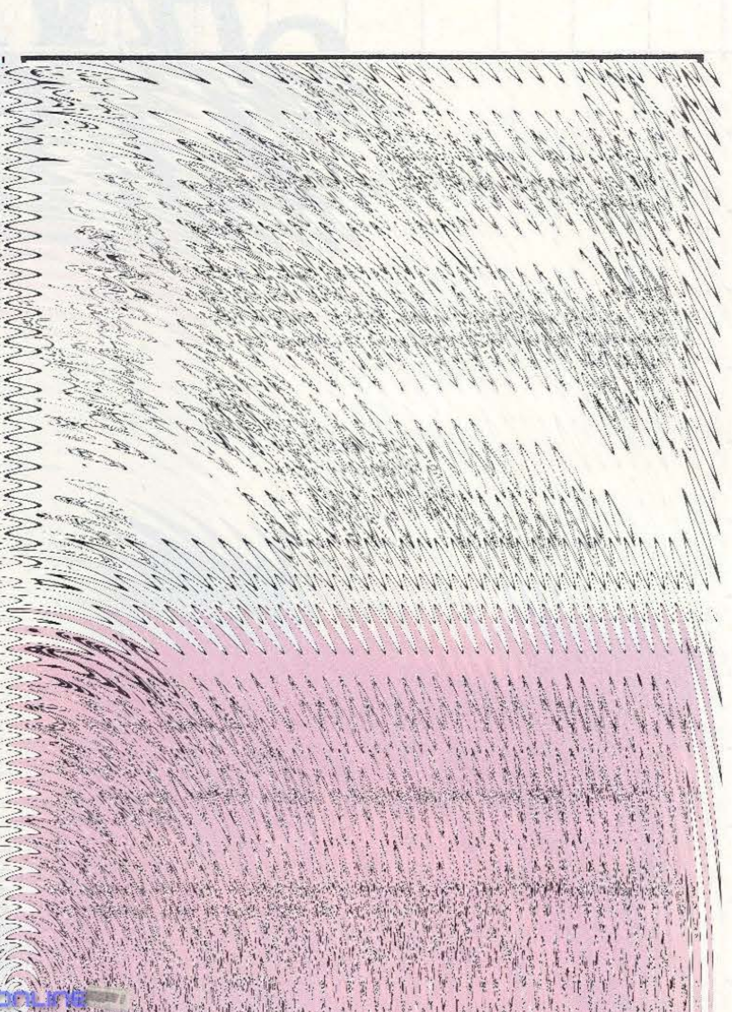
FUNDGRUBE ★ FUNDGRUBE ★ FUNDGRUBE ★ FUNDGRUBE

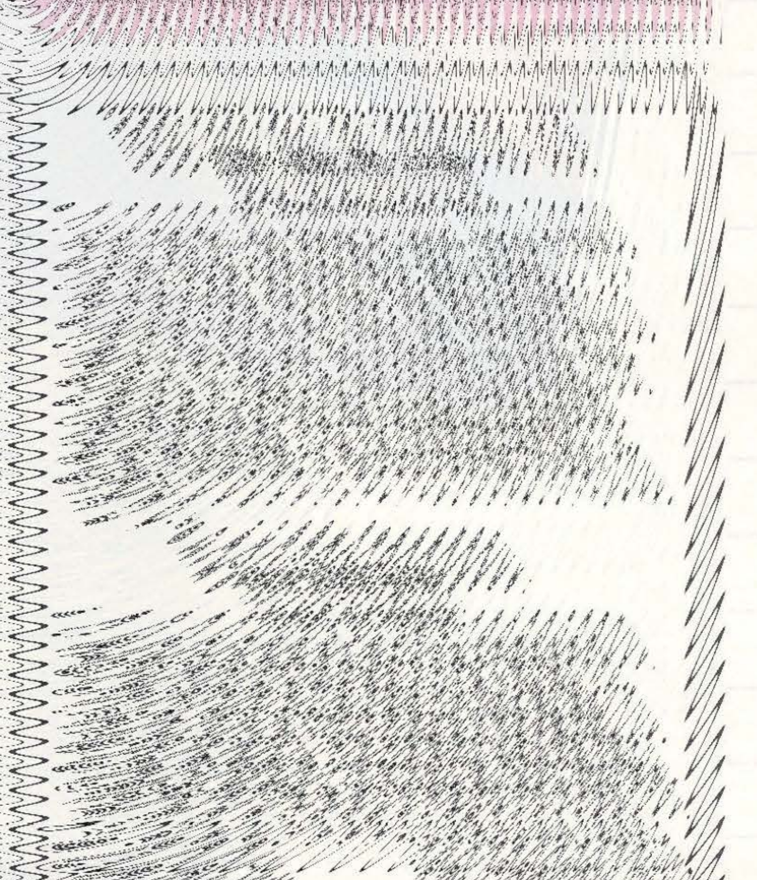
















Einstieg in Simon's Basic

Fast jedes Buch, das sich an den Computerneuling wendet, nimmt für sich in Anspruch, verständlich, stufenweise aufbauend und lerngerecht zu sein. Dieses Buch ist es. Der Autorin ist es gelungen, ihre Erfahrungen aus dem Bereich der Lernpsychologie in das Buch »Einstieg in Simon's Basic« sinnvoll einfließen zu lassen. Das Ergebnis ist verblüffend, denn vom Vorgang des Lernens beziehungsweise Verstehens merkt der Leser nichts. In verständlichen Erklärungen und vielen hilfreichen Zeichnungen sind die Informationen zum schnellen Erlernen des Simon's Basic untergebracht. Dabei wird nicht darauf verzichtet, den Leser zum ständigen Ausprobieren anzuhalten.

Wesentlicher Bestandteil dieses Buches ist es, dem Leser das Wissen auf möglichst vielen Wegen zu übertragen. Die Autorin nennt diese Methode »gehirngerechte« Aufbereitung des Lernstoffes. So wird beispielsweise viel Wert darauf gelegt, die geschriebenen Worte mit Bildern zu unterlegen. Der Computer eignet sich hier als Hilfsmittel hervorragend, denn jedes der kleinen Beispielprogramme wird nicht nur erklärt und abgebildet, sondern kann auch gleich ausprobiert werden. Die Funktion eines Befehls in Simon's Basic zu lesen, im Buch abgebildet zu sehen und auch sofort auszuprobieren ist in der Regel mit »gelernt« gleichzusetzen. Wer also das Buch neben dem Computer liegend, Stück für Stück ausprobiert, wird bald das Gefühl haben, immer schon in Simon's Basic firm gewesen zu sein. Die erreichte Lerntiefe ist so groß, daß auch nach längeren Pausen noch große Teile des Wissens vorhanden sind (Radfahren verlernt man ja auch nicht). Die Lerninhalte des Buches haben ihren Schwerpunkt in den Grafikbefehlen des Simon's Basic. Gleichwohl werden natürlich auch die anderen Befehle erläutert sowie eine große Anzahl nützlicher Hinweise gegeben, die nicht nur mit Simon's Basic anwendbar sind. Das Buch »Einstieg in das Simon's Basic« stellt die augenblicklich gelungenste Anleitung zum Selbststudium dieser Basicerweiterung dar. Der echte Profi wird aber, obwohl uninteressanter zu lesen, sicherlich ein Buch im herkömmlichen Stil verwenden. Bleibt zu bemerken, daß alle Programme im Buch, bis auf das erste, bei dem die Farbgebung verändert werden muß, fehlerfrei funktionieren. Als Sonderservice werden übrigens die wichtigsten Programme aus dem Buch und einige längere nicht abgedruckte Programme auch auf einer Diskette (68 Mark) angeboten. Es ist zu hoffen, daß diese Methode Wissenswertes zu vermitteln auch in zukünftigen Büchern vermehrt anzutreffen ist.

(Arnd Wängler)

Vera F. Birkenbihl, Einstieg in Simon's Basic, 201 Seiten, IWT-Verlag GmbH, Vaterstetten bei München, ISBN 3-88322-056-6, Preis 44 Mark.

Das Musikbuch zum Commodore 64

Mittlerweile scheint es sich auch in den oberen Etagen der Soft- und Hardwareindustrie herumgesprochen zu haben, daß zunehmend mehr Computerfreaks mit wachsender Begeisterung auf ihren Computern Musik machen. Es ist auch wahrlich erstaunlich, was für Möglichkeiten sich hier mittlerweile auftun.

Besonders der Commodore 64 entwickelt sich, zumindest hier bei uns in »Good old Germany«, zum absoluten Renner, was die Beliebtheit bei Computermusik-Freaks angeht. So verwundert es auch nicht, daß, zwar relativ spät, dafür aber gleich von zwei Verlagen fast gleichzeitig, Musikbücher zum Commodore 64 erscheinen.

Die inzwischen schon sehr umfangreiche Data Becker-Bibliothek zum Commodore 64 wurde um »Das Musikbuch zum Commodore 64« erweitert. In altbekannter Data Becker-Aufmachung werden hier, frisch aus dem Computerdrucker und gänzlich ohne schmückendes Beiwerk, viele wichtige Details für den ernsthaften Anwender präsentiert. Als Autor gewann man Thomas Dachsel, der, selbst Musiker, auch das Musikprogramm »Synthimat« entwickelt hat.

Neben einer kurzen Einführung in die Computermusik enthält das Buch auch ein kleines »Lexikon zur Computermusik« und sehr viele Listings von teilweise so bekannten Songs wie »Let it be«, »Hey Jude« und »Yesterday« von den Beatles. Anhand dieser Listings erklärt Dachsel die Feinheiten von einbis schließlich dreistimmigen Basic-Musikprogrammen, wobei natürlich Grundkenntnisse der Programmierkunst vorausgesetzt werden. Anschließend folgt ein Kapitel über den verwendeten SID (Sound Interface Device Chip) mit ausführlicher Beschreibung aller Sound-Register und so grundlegender Dinge der Synthesizertechnik wie Oszillator, Filter, ADSR, Synchronisation und Ringmodulation. Nachdem man dann die Grenzen der Musikprogrammierung in Basic erfährt, macht das »Musikbuch zum Commodore 64« schließlich auch nicht Halt vor der Assembler-Programmierung von Musikstücken. Im Anhang finden sich Hinweise zum Anschluß des C 64 an die Stereoanlage und zur Kopplung mit zusätzlicher Hardware.

Alles in allem ein lesenswertes Buch für alle, die vorhaben, ernsthaft mit dem C 64 Musik zu machen.

(ai)

Thomas Dachsel: Das Commodore Musikbuch, 208 Seiten, Data Becker GmbH 1984, ISBN 3-89011-012-6; Preis 39 Mark

Commodore 64 Musik-Buch

Fast gleichzeitig erschien im Birkhäuser Verlag, Basel, in der Reihe »Computershop« mit dem Band Nr. 14 die deutschsprachige Version des bereits 1983 in Boston erschienenen »The Commodore 64 Music Book«. Die Autoren James Vogel und Nevin B. Scrimshaw haben sich viel Mühe mit der grafischen Gestaltung gegeben, viele Bilder von Musikmaschinen aus grauer Vorzeit lockern die gut gegliederte Lektüre immer wieder auf. Dieses Buch wendet sich mehr an die Einsteiger, mit weniger oder gar keiner Programmiererfahrung. Es beginnt mit der Erklärung von Schall und Schwingung, Frequenz, Tonhöhe, Kurvenformen, ADSR, Filter, Ringmodulation und Synchronisation, so leicht verständlich dargeboten, daß man praktisch ohne Kenntnisse des Programmierens beziehungsweise der Musik zu lesen beginnen kann. Im zweiten Kapitel lernt man mit POKEs umzugehen und dem C 64 erstmals Töne zu entlocken. Hat man das alles intus, geht's ans Programmieren ein- oder mehrstimmiger Melodien mittels READ/DATA-Anweisungen. Abschnitt 5 führt in die Welt der »Magic Effects«. Sirenengeheul, Raketenstart und Lokomotive — der Einsatz von Ringmodulation und Synchronisation, von Filter und Frequenzmodulation wird vor Ort erprobt. Im vorletzten Kapitel eine Einführung in die Musikprogrammierung mit dem Array-Format und schließlich programmiert man eine Orgeltastatur auf seinem C 64 — Echtzeitmusik! Sehr nützlich: Eine herausnehmbare, plastifizierte Tafel, mit einer Tabelle der SID-Tonwerte und den Soundregistern des SID-Chips, die neben den C 64 gelegt viel mühsames Blättern in einem Buch erspart.

Das »Commodore 64 Musik-Buch« ist eine sehr leicht verständliche Einführung in die musikalischen Fähigkeiten des C 64 und vor allem all denen zu empfehlen, die sich noch nicht zu den absoluten Profiprogrammierern zählen.

(ai)

James Vogel und Nevin B. Scrimshaw: Commodore 64 Musik-Buch, 131 Seiten, Birkhäuser-Verlag (Computer Shop Bd. 14) 1984, ISBN 3-7643-1590-3; Preis 29,80 Mark

(Teil 1)

Die meisten Leser haben auf den Mitmach-Karten den Wunsch nach einem Maschinensprachkursus geäußert. Voilà, hier ist er. Dieser Kurs kann kein Buch über Maschinensprache ersetzen, er wird Ihnen jedoch helfen, diese Sprache wesentlich leichter zu verstehen. Sowohl der 6502-Prozessor im VC 20 als auch der 6510-Prozessor im C 64 werden behandelt.

Assembler ist keine Alchimie

Vermutlich hat es Ihnen auch schon ab und zu in den Fingern gejuckt, wenn Sie von Wunderdingen gelesen haben, die man per Maschinensprache mit dem Computer machen kann. Vielleicht haben Sie sogar schon mal nichtsahnend angefangen einzutippen, was Sie als Assemblerlisting sahen. Doch schon nach dem ersten »C000 LDA # $00« und RETURN weigerte sich der Computer mit einem lapidaren »SYNTAX ERROR«. Wieso, werden Sie sich gefragt haben, das ist doch nun die Sprache unserer Maschine, nämlich Maschinensprache, was habe ich falsch gemacht?

Dann sind Sie sicherlich mal auf diese merkwürdigen Basic-Programme gestoßen, in denen ein langer Wurm von DATA-Zeilen mit einem kleinen FOR..NEXT..POKE-Kopf vorne und einem SYS-Schwanz hinten enthalten ist, und die man Basic-Lader nennt. Sie haben fleißig Zahlen eingetippt — das ganze hoffentlich sofort abgespeichert-, vorschriftsmäßig mit dem SYS-Befehl gestartet und auf einen scheintoten Computer geschaut, der nur noch durch Aus- und Einschalten wiederzubeleben war. Wenn Sie dann nach langer Fehlersuche den irrtümlich eingetippten Punkt durch ein Komma ersetzen, (oft finden Sie auch keinen Fehler, denn bei langen DATA-Sequenzen schlägt der Druckfehlerteufel mit Vorliebe zu), werden Sie sich gefragt haben, warum in aller Welt dieses kleine Mißgeschick den ganzen Computer abstürzen läßt. Sie merken vermutlich schon, daß mir das alles und noch mehr (worüber ich schamhaft schweige) passiert ist. Die Konsequenz war, daß ich los ging, um ein schlaues Buch zu erwerben. Aber merkwürdig, damals tauchte der Begriff »Maschinensprache« in keinem Titel auf. Irgendwann begriff ich, daß Assembler und Maschinensprache irgend etwas miteinander zu tun haben.

Aber da fing das ganze Elend erst richtig an: Da gab es 6502-, Z80-, 8080-, 8085-, 6800-Assembler, da waren irgendwelche Schaltpläne, anscheinend, wie man was hinlötet- für mich als Nichtelektroniker eine Art moderner Kunst-, da war von CPU, Bussen, negativen Flanken, Zweiphasentakten die Rede.

Ich habe mich furchtbar geärgert über die Geheimsprache, die es dem Uneingeweihten verwehrt, etwas zu verstehen. Seither hat sich einiges verändert. Die Geheimnisse sind keine mehr und ich werde Ihnen in dieser Serie ohne verschlüsselte Sprache die magischen Zirkel der Assembler-Alchimisten offenbaren. Heute gibt es auch Bücher über »Maschinensprache auf dem Commodore 64« und es sei Ihnen angeraten, ruhig auch das eine oder andere durchzuarbeiten. Sie werden allerdings feststellen, daß die meisten davon gerade dort aufhören, wo es anfängt, spannend zu werden: Bei der Benutzung von Routinen des Betriebssystems und des Interpreters. Deswegen soll der Schwerpunkt dieser Serie anders liegen:

Wir werden das notwendige Grundwissen über die Hardware nur ganz knapp behandeln, dann das Vokabular des 65xx-Assembler kennenlernen. Den Hauptteil der Serie verbringen wir aber mit Dingen, über die es kaum Literatur gibt, nämlich wie man für eine Unzahl von Programmieraufgaben nicht nochmal das Rad erfinden muß, weil es schon längst in unserem Computer existiert.

Bevor wir loslegen, will ich Ihnen noch etwas Literatur empfehlen:

a) Wenn wir über Speicheraufbau, das binäre und das hexadezimale Zahlensystem reden, werde ich voraussetzen, daß Sie die Serie »Reise durch das Wunderland der Grafik« gelesen haben, die in dieser Zeitschrift in den Folgen 1 und 2 (Ausgaben 4/84 und 5/84) diese Themen grundlegend behandelt hat.

b) Als Nachschlagebuch sehr wertvoll ist das Buch von Rodney Zaks: Programmierung des 6502. Das ist gewissermaßen ein Klassiker.

c) Später wird Ihnen dieses Buch fast unentbehrlich vorkommen: R. Babel, M. Krause, A. Dripke: Systemhandbuch zum Commodore 64 (und VC 20), München 1983

Weitere Literaturempfehlungen werde ich Ihnen von Fall zu Fall geben. Gerade zu unserem Computer erscheint fast jeden Monat ein neues Buch und es ist nicht einfach, die Spreu vom Weizen zu trennen.

Einige Begriffsklärungen

Zunächst einmal muß ich Sie enttäuschen: Ich glaube kaum, daß Sie mit Ihrem Computer je einmal in Maschinensprache verkehren werden! Maschinensprache, das ist die einzige, die der Computer direkt versteht, das sind vorhandene oder nicht vorhandene Stromimpulse oder Magnetisierungszustände, die bei unserem Computer durch 8-Bit-Binärzahlen auszudrücken sind. Was wir mit unserem Computer reden werden ist Assembler. Mit dem Computer sprechen soll heißen: Mit dem Gehirn unseres Computers, dem Prozessor, oft auch CPU (von Central Processing Unit = Zentraler Arbeitsbaustein) genannt, verkehren, also ihm Befehle zu geben. Solche CPUs werden bei verschiedenen Firmen hergestellt, sind daher unterschiedlich aufgebaut und auch unterschiedlich ansprechbar. Ein weit verbreiteter Prozessortyp ist der 6502, der das Gehirn des C 64 und auch des VC 20 ist. Genau genommen ist das Gehirn des C 64 allerdings der 6510, ein dem 6502 fast identischer Prozessor. Auf den kleinen Unterschied werden wir noch zu sprechen kommen. Beide (6502 und 6510) sind in 6502-Assembler zu programmieren und wenn wir diese Sprache sprechen, sind für uns alle 6502-Computer zugänglich: Commodore, Apple, Atari und einige an-

dere. Nun wissen Sie aber immer noch nicht, was Assembler eigentlich ist. Das englische Wort »assemble« heißt auf deutsch etwa montieren, zusammenstellen. Es handelt sich also um eine Programmiersprache und weil sie sehr eng am Computer orientiert ist, spricht man von einer »maschinenorientierten« Programmsprache im Gegensatz zu »problemorientierten« Programmsprachen wie Basic, Pascal, Cobol etc., die — so sollte es jedenfalls sein — auf jedem Computertyp gleich aussehen.

Ein Assembler ist aber noch etwas anderes, nämlich ein Software-Instrument, das einen in Assembler geschriebenen Befehl in die Maschinensprache übersetzt. Man spricht vom Vorgang des Assemblierens. Das umgekehrte leistet ein Disassembler, welcher uns Maschinensprache durch Rückübersetzung lesen hilft. Um die Verwirrung noch etwas zu steigern, sage ich Ihnen auch noch, was ein Monitor ist. In diesem Zusammenhang ist kein Bildschirmgerät damit gemeint, sondern ebenfalls ein Software-Instrument, das den Einblick in die Register und Speicher des Computers gewährt.

Damit Sie nun den Überblick noch völlig verlieren, sei abschließend zu diesem Sprachenwirrwarr noch erzählt, daß Software-Pakete, die sowohl Assembler als auch Disassembler als auch Monitor enthalten und noch eine Menge anderer brauchbarer Dinge, oft als »Assembler« angeboten werden. Das ist ein alter Trick der Alchimisten, verschiedenen Dingen den gleichen Namen zu geben!

Basic contra Assembler

Um das Nachfolgende deutlich zu machen, schalten Sie bitte Ihren Computer an und tippen die beiden folgenden Programme ein, die beide genau dasselbe tun: Das obere Viertel unseres Bildschirmes mit dem Buchstaben A zu füllen (beim VC 20 ist es die obere Hälfte). Zunächst einmal in Basic:

```
10 FOR I=1024+255 TO 1024 STEP-1
20 POKE I, 1:POKE I+54272,14
30 NEXT I
```

Für den VC 20 (Grundversion und 3-KByte-Erweiterung) ist zu setzen: Anstelle von 1024 jetzt 7680, statt 54272 jetzt 30208 und statt 14 die 6. Wenn Sie mehr als die 6,5 KByte im VC 20 haben, dann setzen Sie statt 1024 jetzt 4096, statt 54272 jetzt 34304 und ebenfalls statt 14 die 6. Das Programm braucht 55 Byte + 7 Byte für die Variable I, macht zusammen 62 Byte Speicherplatz. Es geht ganz schnell und wenn Sie es schaffen, können Sie ja mal mitstoppen, wie lange es von RUN bis READY braucht: Zirka 4 Sekunden.

Jetzt dasselbe in Assembler. Weil wir aber noch nicht soweit sind, erst mal als Basiclader, der uns das Programm in den Speicher bringt (wir kommen dazu gleich noch). Geben Sie also NEW ein und dann:

```
10 FOR I=7000 TO 7000+16
20 READ A:POKE I,A:NEXT I :END
30 DATA 160,255,162,14,169,1,153,255,
3,138,153,255,215,136,208,244,96
```

Beim VC 20 geben Sie bitte statt der 14 (Zeile 30,4.Zahl) eine 6 ein. Starten Sie den Basiclader mit RUN und nach dem READY geben Sie NEW und CLR ein: wir brauchen ihn den Basiclader nicht mehr. Ab Speicherstelle 7000 steht jetzt unser Assemblerprogramm als Maschinencode. Daß es wirklich dasselbe tut wie das Basicprogramm erfahren Sie durch SYS 7000. Da hatten Sie vermutlich gar keine Zeit mehr, auf die Stoppuhr zu drücken! (5,4 Millisekunden etwa dauert das ohne die Zeit, die der Basicinterpreter für den Befehl SYS benötigt). Außerdem braucht das Programm 17 Byte Speicherplatz.

Genau das ist es, was die Assemblerprogrammierung so reizvoll macht: Der Speicher faßt mehr an Programm und die Ausführung des Programmes geht fast 1000mal so schnell! Dazu kommen natürlich noch einige andere Kriterien, denn viele Probleme sind zum Beispiel in Basic nicht lösbar, sondern nur mit dem vielseitigeren Assembler.

Unser Computer ist darauf vorbereitet, daß wir ihn in Basic ansprechen. Er enthält im Normalfall sofort nach dem Einschalten ein stets präsentes Übersetzungsprogramm, den Interpreter, welcher unsere Basicanweisungen für ihn verständlich interpretiert. Auch das ist ein Unterschied zu Assemblerprogrammen: Ist ein solches Programm erst einmal assembliert (also als Maschinensprache im Speicher vorhanden), braucht man kein Übersetzungsprogramm mehr. Basicprogramme dagegen müssen bei jedem Durchlauf von vorne bis hinten ständig übersetzt werden, sie laufen nicht ohne vorhandenen Interpreter. Wie so ein Interpreter im Prinzip arbeitet und was ihn von einem sogenannten Compiler unterscheidet, sollten Sie mal in der April-Ausgabe der Zeitschrift 64'er im Artikel ab Seite 110 von M. Törk über seinen Strubs-Precompiler nachlesen.

Dort sehen Sie dann auch, daß ein Compiler zwar ein Basicprogramm enorm beschleunigen kann, aber bei weitem nicht an die Geschwindigkeit reiner Assemblerprogramme heranreicht, vom Speicherplatzbedarf ganz zu schweigen.

Wie sag ich's meinem Computer?

Leider haben weder der C 64 noch der VC 20 einen Assembler implementiert. (Sie merken, daß jetzt von dem Software-Paket die Rede ist!). Es gibt einen etwas mühseligen Weg, dieses Handicap zu umgehen: Den Basiclader. Wie ist also der Weg, mit einem solchen Lader eigene Maschinenprogramme in den Computer zu bekommen?

a) Erstellen des Assemblerprogrammes. Das zu lernen ist die Hauptaufgabe in der Serie. Das Ergebnis wird eine Kette von Befehlen sein, zu denen zum Beispiel der Befehl RTS gehört.

b) Jedem Befehl in Assembler entspricht in Maschinensprache ein Binärcode in einer Speicherstelle. Diese Codes sind in Listen nachschlagbar: RTS entspricht dem Binärcode 0110 0000.

c) Der Code muß in eine Speicherstelle eingegeben werden. Das geschieht von Basic aus mit dem POKE-Befehl. Weil aber Basic keine Binärzahlen kennt, muß der Code ins Dezimalsystem umgerechnet werden. Glücklicherweise sind in den Tabellen meist schon die Codes als Dezimal- oder wenigstens als Hexadezimalzahlen enthalten. RTS ist dezimal 96 (oder hex.60, das auch $ 60 geschrieben werden kann). Man POKEt nun an die richtige Adresse den Wert 96, also zum Beispiel POKE 7016,96

d) Auf diese Weise wird Byte für Byte in der Programmabfolge verfahren. Das reine POKEn geschieht dann eben in der Form wie im oben gezeigten Basiclader. Mühsam, mühsam! Auch kann man leider nur mit dem PEEK-Kommando nachsehen, was denn nun im Speicher steht (PEEK (7016) gibt uns den Wert 96, entsprechend RTS).

Ein anderer Weg ist der Kauf eines Assembler-Moduls oder einer entsprechenden Programm-Kassette oder Diskette. Sehr preiswert ist es sicherlich, in Fachzeitschriften Umschau zu halten nach abge-

druckten Assemblern. Sehr gut finde ich beispielsweise den von U. Roller in Chip Nummer 1 (1984), aber auch im 64'er wird während des Verlaufs dieser Serie ein gut verwendbares Programm vorgestellt.

Assembler (das Software-Paket) gibt es in den unterschiedlichsten Ausführungen. Es gibt beispielsweise Direkt-Assembler, die jede Programmzeile sofort nach dem RETURN assemblieren, aber auch 2-Pass-Assembler, bei denen das erst nach Abschluß des Programms insgesamt durch einen Befehl (zum Beispiel ASSEMBLE) geschieht. Bei einigen kann man (ähnlich wie bei Basic mit REM) Kommentare anfügen, bestimmten Programmstellen Namen geben (sogenannte LABELS), ganze Programmabschnittte mit einem Merknamen aufrufen (sogenannte MAKROS) und so weiter. Was Sie für sich bevorzugen, bleibt Ihnen natürlich überlassen. Die in dieser Serie geschriebenen Programme werden auf diese schönen Erleichterungen verzichten, es wird sozusagen der nackte Assembler verwendet und solange in dieser Zeitschrift noch kein Software-Paket dafür veröffentlicht ist, werde ich jedes Programm, das wir zusammen entwickeln, als Basiclader angeben. Was Sie aber außer dem reinen Assembler noch brauchen, ist ein Disassembler und ein Monitor (ich habe schon erklärt, welchen ich meine), damit wir unseren Computer (fast) immer im Griff haben.

Wie funktioniert unser Computer?

Weil das Programmieren in Assembler einen viel engeren Kontakt zu technischen Einzelheiten unseres Computers erfordert, ist es notwendig, ein wenig über diese Innereien und ihre Funktion zu wissen. Sehen Sie sich dazu bitte das Bild 1 an.

Da sehen wir zunächst unseren Mikroprozessor, der meist eine Menge Funktionen in sich vereinigt (dazu kommen wir noch). Im Prinzip ist das unsere CPU (Zentraler Arbeitsbaustein). Der Prozessor steht über eine Reihe von Leitungen mit dem Rest des Computers in Verbindung. Diese Leitungen werden im Fachjargon BUSSE genannt. Da ist zunächst einmal der sogenannte Adreßbus, auf dem 16-Bit-Adressen transportiert werden, die der Prozessor erzeugt, und die die Herkunft oder auch das Ziel von Daten festlegen, die über den Datenbus laufen. Dieser kann 8-Bit-Daten transportieren, und zwar schreibend oder lesend, also zum Beispiel vom Prozessor zum RAM (schreibend), vom RAM zum Prozessor (lesend) und so weiter. Außerdem gibt es da noch einen Steuerbus, der verschiedene Synchronisationsaufgaben durchführen hilft. Links vom Prozessor ist ein Taktgeber angedeutet. Damit nichts durcheinander kommt, läuft alles im Computer sozusagen im Gleichschritt. Diese Uhr ist gewissermaßen der Trommler, den Sie vielleicht von den alten Ruder-Galeeren kennen. Dann sehen Sie rechts einen ROM-Bereich, also einen Nur-Lese-Speicher (Read Only Memory). Daß man hier nur herauslesen kann, ist durch den Pfeil zum Datenbus gekennzeichnet. Doppelpfeile finden wir aber beim RAM (Random Access Memory), einem Speicher für beliebigen Zugriff, also lesend und schreibend, und bei den Ein- und Ausgabebausteinen, die den Kontakt des Computers mit der übrigen Welt erlauben, also auch mit uns. Dieses Aufbauprinzip finden wir bei allen 8-Bit-Computern.

Das Innenleben eines Mikroprozessors

Um es gleich nochmal zu sagen: Was hier erzählt wird, ist nicht dazu geeignet, Elektronik-Freaks den totalen Durchblick zu geben. Wenn Sie das aber gerne möchten, dann sehen Sie sich zum Beispiel die Blockschaltbilder an im »Programmer's Reference Guide« für den Commodore 64 auf Seite 404 oder im »MOS-Hardware-Handbuch« auf Seite 34. Auch Rodney Zaks in dem anfangs schon erwähnten Buch »Programmierung des 6502« ist zu empfehlen. Er hat sich viel Mühe gegeben, sich verständlich auszudrücken. Mir kommt es nur auf den allgemeinen Überblick an. Den sollen Sie bekommen, wenn wir uns jetzt zusammen Bild 2 betrachten.

Da sehen Sie zunächst als Herzstück des Prozessors, die ALU (Arithmetik Logical Unit), also den arithmetisch-logischen Baustein. Die ALU hat die Fähigkeit, Rechenoperationen auszuführen mit Daten, die sie über den Datenbus und normalerweise vom Akkumulator erhält. Das Ergebnis wird ebenfalls im Akkumulator abgelegt (daher auch der Name: von akkumulieren, etwa ansammeln). Der Akkumulator ist das Register, das uns als Programmierer am häufigsten beschäftigen wird. Er ist die Sammel- aber auch die Verteilerstelle für fast alle Daten, die wir hin- und herschieben wollen. Sowohl der Akku (so werde ich, mit der Hoffnung auf Ihr wohlwollendes Verständnis, künftig bezeichnen) als auch alle anderen Register, das heißt, die höchste Zahl, die darin bearbeitet werden kann, ist 255 (binär 1111 1111). Nahezu ebenso oft wie den Akku werden wir die beiden sogenannten Index-Register X und Y benutzen. Warum man sie Index-Register nennt, sehen Sie noch im Verlauf der Serie. Als nächstes zum Prozessor-Statusflaggen-Register (hier P genannt). Man findet hier angezeigt, ob eine Rechenoperation ein negatives Ergebnis hatte oder ob eine Null aufgetaucht ist oder ob ein Übertrag stattgefunden hat. Auch dieses Register wird uns noch häufig begegnen. Das Stapelregister, auch Stackpointer (Stapelzeiger) genannt, gibt uns Auskunft über den Füllungsgrad eines 256 Byte großen speziellen Speichers, der vom Prozessor direkt verwaltet wird. Auch damit werden wir noch oft zu tun haben. Schließlich kommen wir zur vorhin erwähnten Ausnahme, zum Programmzähler (PCL, PCH). Das ist ein 16-Bit-Register, das sich aus zwei 8-Bit-Registern (PCL für das LSB und PCH für das MSB) zusammensetzt und daher alle 65535 Speicherplätze ansprechen kann. Hier ist immer die Adresse des nächsten abzuarbeitenden Befehls enthalten.

Ich will an dieser Stelle nicht in die Einzelheiten der Befehlsabarbeitung einsteigen (das können Sie auch bei Rodney Zaks nachlesen, wenn Sie's genau wissen wollen). Es soll nur gesagt sein, daß sich die Verarbeitung in drei Schritte unterteilen läßt:

a) den nächsten Befehl holen
b) den Befehl decodieren
c) den Befehl ausführen

Zu c) ist noch zu sagen, daß es Befehle gibt, die der Prozessor ohne weitere Angaben ausführen kann. Für andere müssen erst noch weitere Daten aus dem Speicher geholt oder dort abgelegt werden. Deswegen brauchen die Befehle unterschiedliche Zeiten zur Ausführung. Die Zeit wird als Anzahl von sogenannten Taktzyklen in den Befehlstabellen angegeben. Unser Computer hat eine Taktfrequenz von rund 1 MHz, was bedeutet, daß ein Taktzyklus etwa eine Mikrosekunde (10^{-6} Sekunden) dauert. Auf diese Weise wurde die Zeitdauer für unser kleines Demonstrationsprogramm zu ►


64er online

Anfang berechnet. Auch das werden Sie noch lernen.

Der Speicher unseres Computers: Eine Straße mit 65536 Hausnummern

Tabelle 1. Basic-Start- und -Endadressen beim VC 20 mit verschiedenem Speicherausbau

Grundversion	:Basic-Start	4096	Basic-Ende	7679
+3-K-Erweiterung	: —"—	1024,	—"—	7679
+8-K-Erweiterung	: —"—	4608,	—"—	16383
+16-K-Erweiterung	: —"—	4608,	—"—	24575
+24-K-Erweiterung	: —"—	4608,	—"—	32767

Diese Serie ist für den VC 20 und den C 64 geschrieben. Den Speicheraufbau des Commodore 64 finden Sie in der April-Ausgabe dieser Zeitschrift ab Seite 119. Deswegen soll hier nur der des VC 20 gezeigt werden. Man muß beim VC 20 zwei Konfigurationen unterscheiden — sehr zum Leidwesen der Benutzer. In Bild 3 ist die Aufteilung gezeigt, die in der Grund- und der um 3 KByte erweiterten Version vorliegt.

In Bild 4 sehen Sie die Speicheraufteilung, die bei mehr als 6,5 KByte eingestecktem Speicher gültig ist.

Wenn Sie die VC 20 Speicherar-

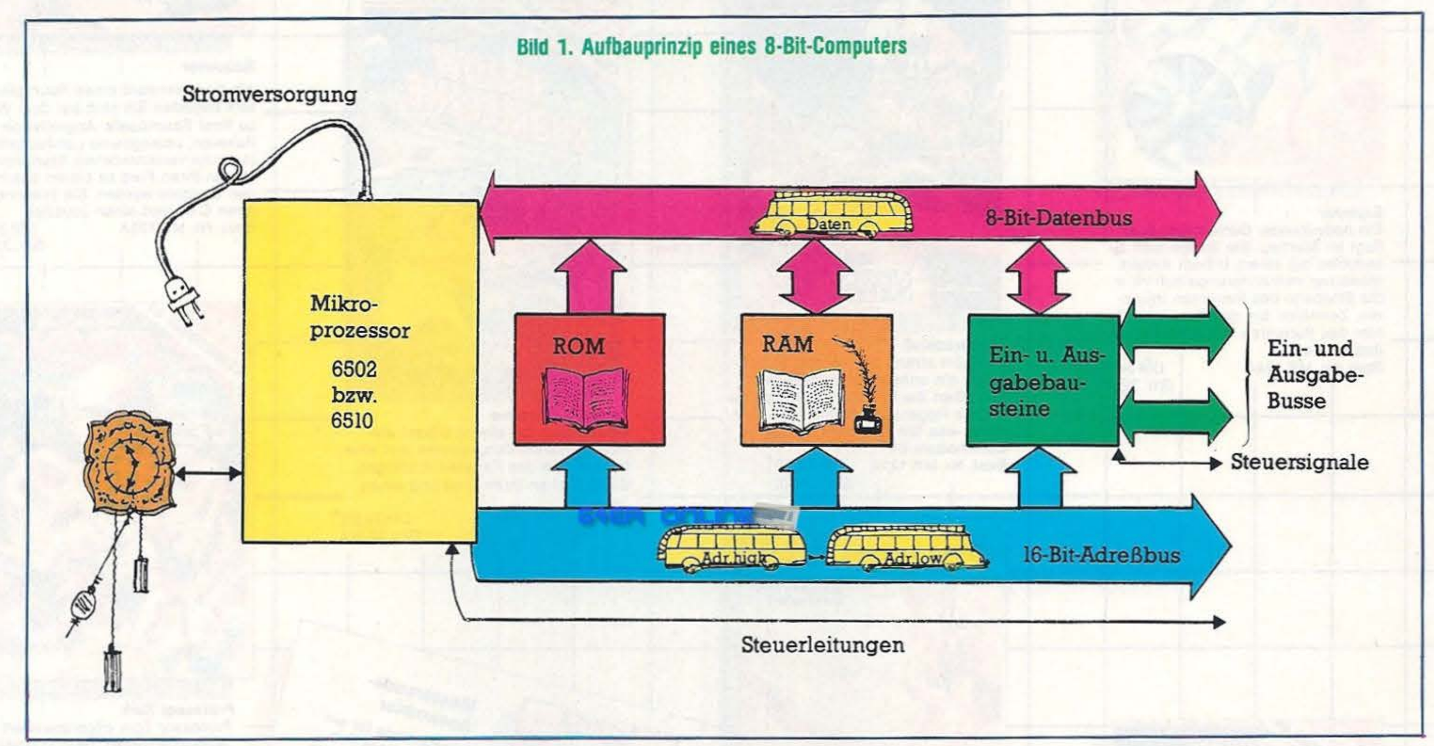


Bild 1. Aufbauprinzip eines 8-Bit-Computers

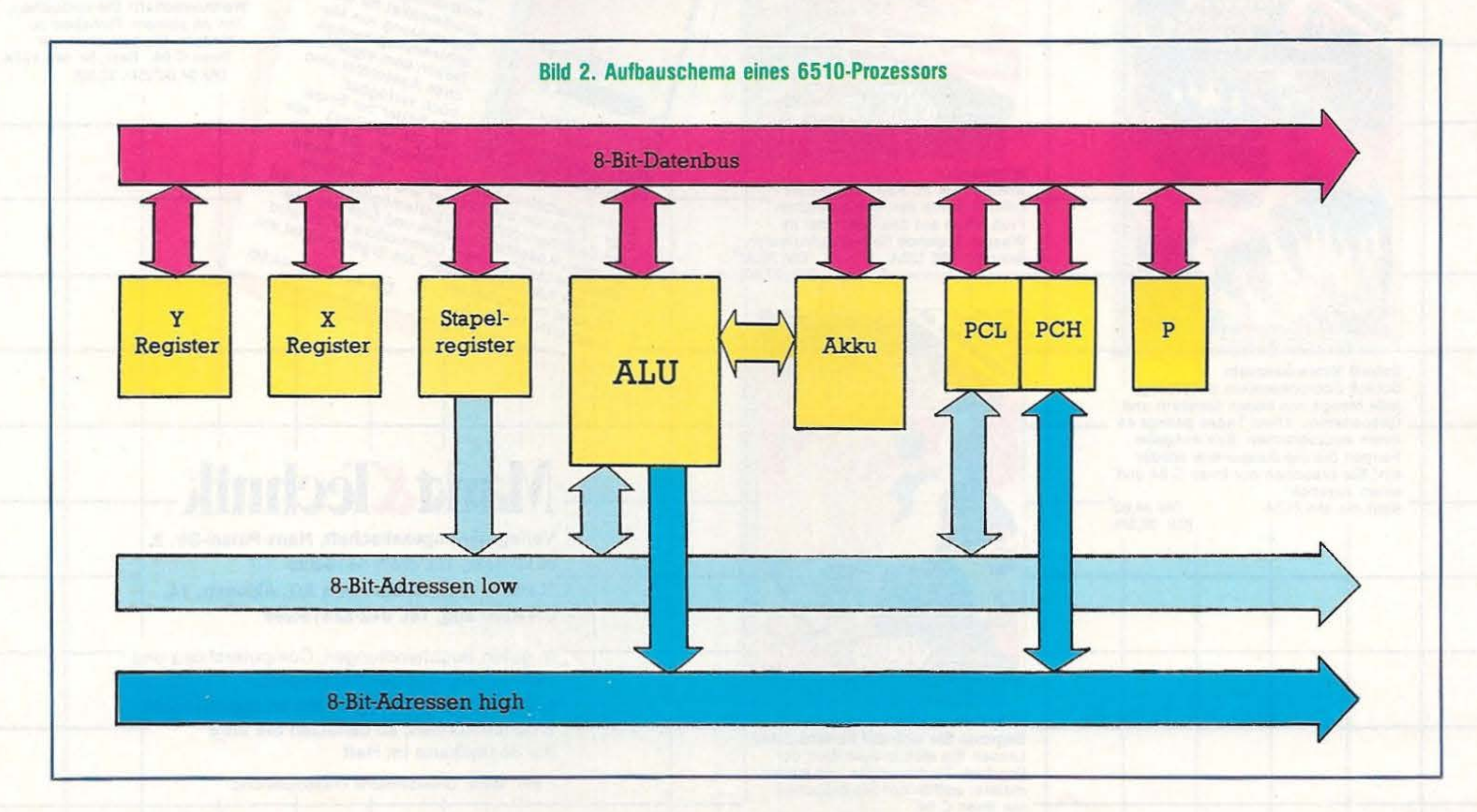


Bild 2. Aufbauschema eines 6510-Prozessors

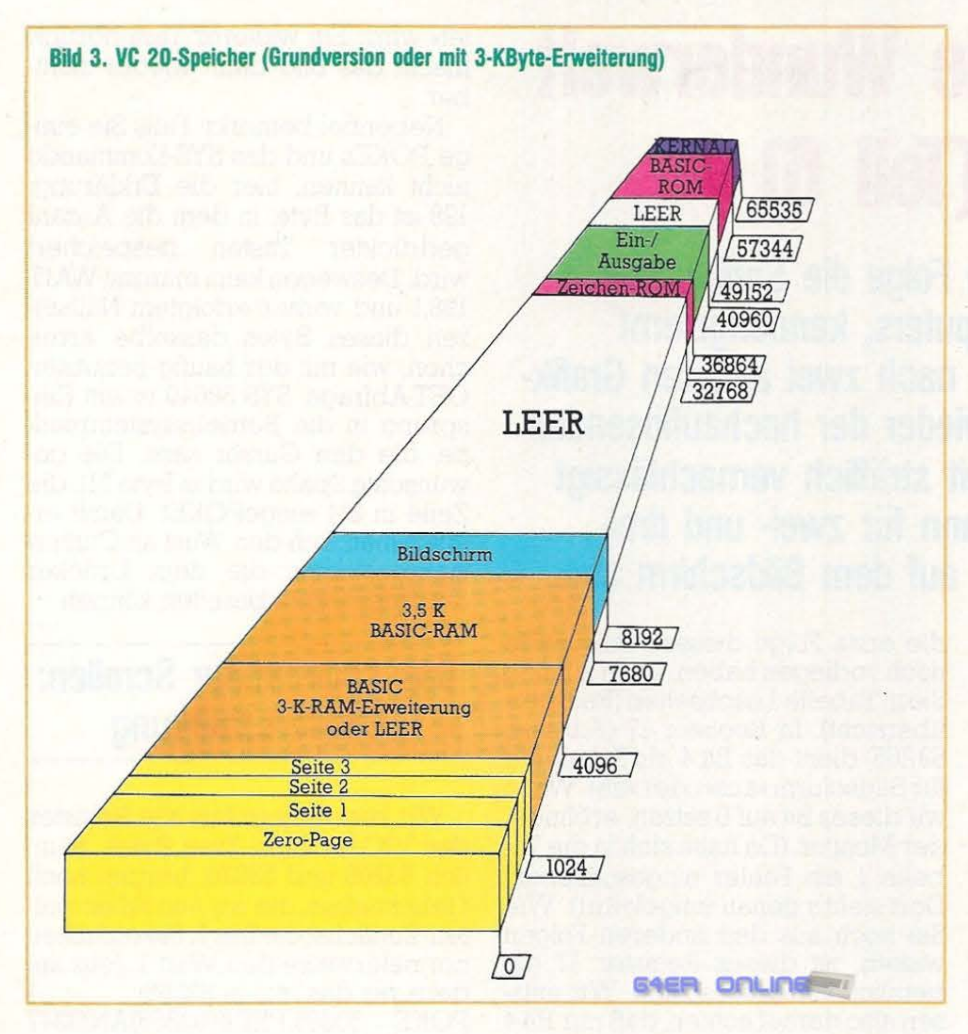


Bild 3. VC 20-Speicher (Grundversion oder mit 3-KByte-Erweiterung)

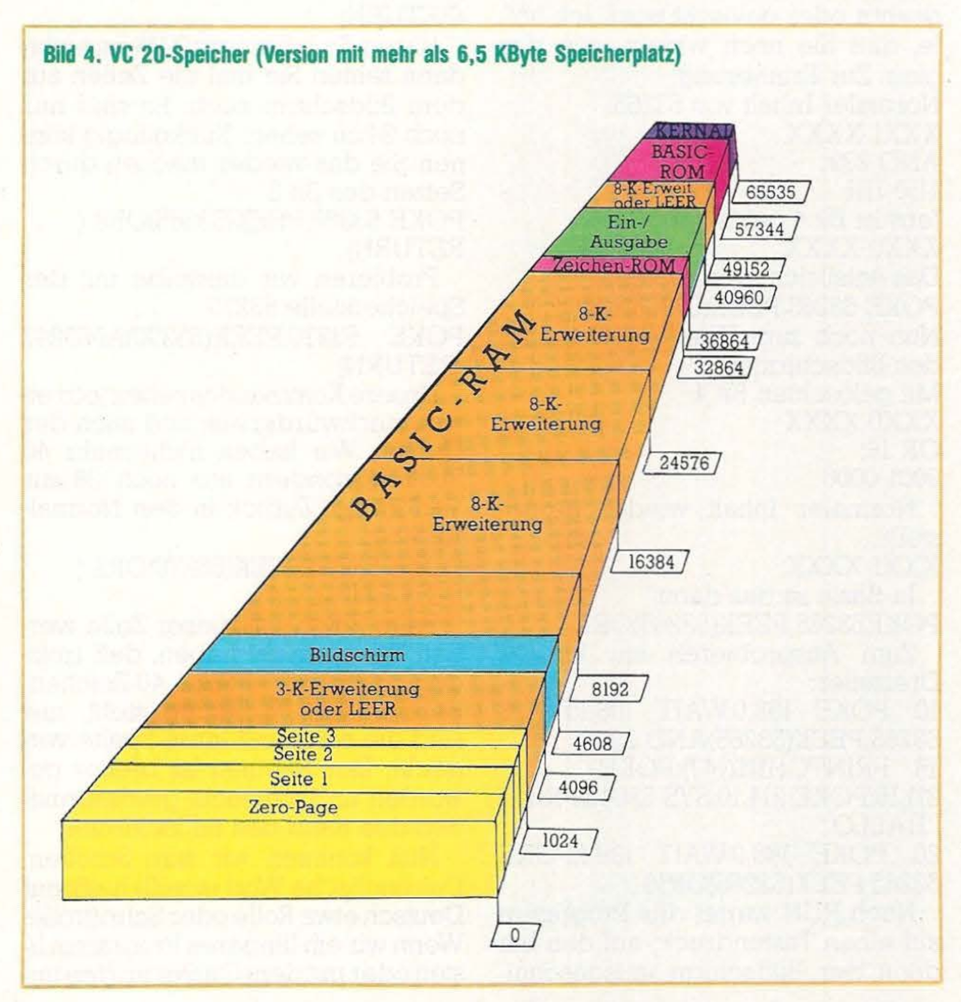


Bild 4. VC 20-Speicher (Version mit mehr als 6,5 KByte Speicherplatz)

chitekturen mit der des C 64 vergleichen, werden Sie eine Reihe von Unterschieden feststellen. Genau besehen gibt es an den wichtigen Punkten aber eine Menge Gemeinsamkeiten! Der VC 20 kennt nur Speicher-Häuser mit Erdgeschoß, im Gegensatz zum C 64, wo manche Bereiche sogar zwei Etagen haben (soll heißen: mehrfach belegt sind). Durch die Eigenart des C 64 aber, im Normalfall das Basic-ROM, die Ein- und Ausgabebausteine und das Betriebssystem eingeschaltet zu haben, kann man ihn eigentlich genauso behandeln wie einen VC 20, bei dem die genannten ROM-Bausteine, — und zwar das Basic-ROM —, um 8 KByte verschoben sind. Die Unterschiede der ROM-Inhalte sind fast vernachlässigbar. Wir werden im Einzelfall darauf zu sprechen kommen. Bei den Ein- und Ausgabebausteinen liegen allerdings größere Unterschiede.

Die Seiten 0 bis 3 (eine Seite oder auch page enthält 256 Byte und man zählt oft auch in diesen Seiten, wenn vom Speicher die Rede ist), sind sich ebenfalls sehr ähnlich und die wenigen Unterschiede werden uns ebenfalls noch beschäftigen. Der Bildschirm liegt bei der Grundversion und der mit der 3-KByte-Erweiterung von 7680 bis 8191, in der Version mit mehr als 6,5 KByte von 4096 bis 4607 und beim C 64 von 1024 bis 2047. Der Bildschirmfarbspeicher liegt — bei gleicher Reihenfolge — von 37888 bis 38399, beziehungsweise von 38400 bis 38911 und schließlich von 55296 bis 56295. Der Basic-RAM-Bereich beginnt beim C 64 im Normalfall bei 2048 und endet bei 40959. Beim VC 20 ist das natürlich wieder von der jeweiligen Erweiterung abhängig (Tabelle 1).

Dies gilt — wie Sie leicht auch aus Bild 4 sehen können — auch dann, wenn zu den 8 KByte/16 KByte/24-KByte-Erweiterungen noch die 3-KByte-Erweiterung und die 'KByte-Erweiterung im hohen Speicherbereich (40960 bis 49151) verwendet werden. Diese letztgenannten Adressenbereiche sind dann gut als geschützte RAM-Bereiche für Maschinensprache zu verwenden, ebenso wie beim C 64 der Speicherabschnitt von 49152 bis 53247.

Damit schließen wir zunächst mal den Hardware-Überblick ab. In der nächsten Folge stelle ich Ihnen dann die ersten Assembler-Befehle vor, und wir beginnen zu programmieren. Sie gehören nun zum 1. Grad der Assembler-Alchimisten.

(Heimo Ponnath/aa)

Reise durch die Wunderwelt der Grafik (Teil 6)

Nachdem wir in der letzten Folge die Sprites, einen Leckerbissen unseres Computers, kennengelernt haben, wenden wir uns — nach zwei anderen Grafikeigenheiten des C 64 — wieder der hochauflösenden Grafik zu, die wir einige Zeit sträflich vernachlässigt haben. Diese nützen wir dann für zwei- und dreidimensionale Darstellungen auf dem Bildschirm aus.

Zunächst aber ein leichtes und ein schwierigeres Thema:

Wenn Sie an Ihrem Computer eine Kassettenstation verwenden, dann kennen Sie das ja zur Genüge. Sie geben irgendeinen Kassettenbefehl ein, der Computer fordert zum Beispiel PRESS RECORD & PLAY. Sobald Sie dieser Aufforderung nachgekommen sind, ist der Bildschirm total leer. Dieses üble Verhalten — wir sehen ja nun einige Meldungen nicht mehr — hat seinen Grund. Die Kassettenoperationen können etwas schneller laufen. Überhaupt beschleunigt ein abgeschalteter Bildschirm einen Programmablauf. Im allgemeinen nützt uns das nicht viel, denn wir reden ja mit unserem Computer per Bildschirm und wenn der Gesprächspartner einen »black-out« hat, kann nicht miteinander verkehrt werden. Manchmal gibt es aber Situationen, wo dieses Abschalten für kurze Zeit ganz angenehm sein kann. Wenn wir zum Beispiel ein kompliziertes Titelbild aufbauen, mögen viele nicht zuschauen. Sondern nach einem leeren Bildschirm soll sofort das fertige Titelbild auftauchen.

Was passiert beim Abschalten des Bildschirmes? Das ist optisch eigentlich gar nicht so aufregend, denn der ganze Bildschirm nimmt die Farbe des Rahmens ein. Er wird also genau genommen nicht abgeschaltet. Geschaltet wird etwas, das wir nicht sehen können. Nämlich eine ganz bestimmte Fähigkeit unseres Prozessors, auf den Systembus anders als im Normalbetrieb zuzugreifen. Diese Art des Zugriffs birgt eine versteckte, mögliche Fehlerquelle: Wenn während Kassettenoperationen Sprites angeschaltet bleiben, können Störungen beim Zugriff auf Daten auftreten. Falls Sie die erste Folge dieser Artikelserie noch vorliegen haben, dann können Sie in Tabelle 1 nachsehen (Registerübersicht). In Register 17 (Adresse 53265) dient das Bit 4 als Schaltbild für Bildschirm »aus« oder »an«. Wenn wir dieses Bit auf 0 setzen, erblindet der Monitor. (Da hatte sich in die Tabelle 1 ein Fehler eingeschlichen: Dort steht's genau umgekehrt). Wie Sie noch aus den anderen Folgen wissen, ist dieses Register 17 ein ziemlich komplexes Ding. Wir müssen also darauf achten, daß nur Bit 4 gesetzt oder gelöscht wird. Ich hoffe, daß Sie noch wissen, wie das ging. Zur Erinnerung:

Normaler Inhalt von 53265:
XXX1 XXXX
AND 239:
1110 1111
Jetzt ist Bit 4 gelöscht:
XXX0 XXXX
Das entspricht dem Befehl:
POKE 53265,PEEK(53265)AND239
Nun noch zum Wiedereinschalten des Bildschirmes:
Mit gelöschten Bit 4:
XXX0 XXXX
OR 16:
0001 0000

Normaler Inhalt wieder hergestellt:
XXX1 XXXX

In Basic ist das dann:
POKE53265,PEEK(53265)OR16.

Zum Ausprobieren ein kleiner Dreizeiler:

```
10 POKE 198,0:WAIT 198,1:POKE 53265,PEEK(53265)AND 239
15 PRINTCHR$(147):POKE 211,15:POKE 214,10:SYS 58640:PRINT "HALLO"
20 POKE 198,0:WAIT 198,1:POKE 53265,PEEK(53265)OR16
```

Nach RUN wartet das Programm auf einen Tastendruck, auf den hin dann der Bildschirm »ausgeschaltet« wird. Ein weiterer Tastendruck macht das Bild dann wieder sichtbar.

Nebenbei bemerkt: Falls Sie einige POKEs und das SYS-Kommando nicht kennen, hier die Erklärung: 198 ist das Byte, in dem die Anzahl gedrückter Tasten gespeichert wird. Deswegen kann man mit WAIT 198,1 und vorher erfolgtem Nullsetzen dieses Bytes dasselbe erreichen, wie mit der häufig benutzten GET-Abfrage. SYS 58640 ist ein Einsprung in die Betriebssystemroutine, die den Cursor setzt. Die gewünschte Spalte wird in Byte 211, die Zeile in 214 eingePOKEt. Damit erspart man sich den Wust an Cursorsteuerzeichen, die dem Drucker Kopfschmerzen bereiten können.

Das sogenannte Scrollen: Bildverschiebung

Wir kennen jetzt fast alle Register des VIC-II-Chips. Zwei Bytes, nämlich 53265 und 53270, bergen noch Geheimnisse, die wir nun lüften wollen. Zunächst die Bits 3. Sie enthalten normalerweise den Wert 1. Jetzt ändern wir das mal in 53265:
POKE 53265,PEEK(53265)AND247 ⟨RETURN⟩

Haben Sie's bemerkt? Wenn nicht, dann zählen Sie mal die Zeilen auf dem Bildschirm nach: Es sind nur noch 24 zu sehen. Rückgängig können Sie das wieder machen durch Setzen des Bit 3:
POKE 53265,PEEK(53265)OR8 ⟨RETURN⟩

Probieren wir dasselbe mit der Speicherstelle 53270:
POKE 53270,PEEK(53270)AND247 ⟨RETURN⟩

Unsere Kommandos sehen jetzt etwas merkwürdig aus und auch der Cursor. Wir haben nicht mehr 40 Spalten, sondern nur noch 38 zur Verfügung. Zurück in den Normalzustand:
POKE 53270,PEEK(53270)OR8 ⟨RETURN⟩

Beim Eingeben dieser Zeile werden Sie bemerkt haben, daß trotzdem noch der ganze 40-Zeichen-Bereich zur Verfügung steht, nur sind die erste und letzte Spalte verdeckt. Der Rahmen ist breiter geworden und versteckt gewissermaßen den Rand des Bildschirmes.

Nun kommen wir zum Scrollen. Das englische Wort »scroll« heißt auf Deutsch etwa Rolle oder Schriftrolle. Wenn wir ein längeres Programm listen oder mit dem Cursor an den un-

teren Bildrand fahren, rollt der C 64 den Bildschirminhalt nach oben. Das ist ein zeilenweises Scrollen. Der VIC-II-Chip gestattet aber auch eine andere Art des Scrollens, die im englischen als »smooth scrolling«, etwa »fließendes« Scrollen bezeichnet wird. Wie Sie sich sicherlich erinnern, ist jedes Zeichen aus 8 Bytes zusammengesetzt. Während beim erstgenannten Scrollen alle 8 Bytes (also ein Zeichen) auf einmal nach oben springen, geht das beim fließenden Scrollen Byte für Byte. Dazu dienen die Bits 0 bis 2 der Register 53265 und 53270. Sehen wir uns das mal in Einzelschritten an. Der Maximalwert in den 3 Bits (0,1,2) kann 7 sein (binär 111). Alle anderen Bits müssen geschützt sein. Normalerweise steht in 53265 in den Bits 0 bis 2 der Wert 3 und an derselben Stelle in 53270 der Wert 0. Folgendes Kurzprogramm soll diese Werte mit jedem Tastendruck verändern:

```
5 PRINT CHR$(147)CHR$(166)
10 FOR I = 0 TO 7
20 POKE 53265,(PEEK(53265)AND
248)OR I
30 GET A$:IF A$=""THEN 30
40 NEXT I
50 POKE 53265,(PEEK(53265)AND
248)OR3
```

Wenn Sie dieses Programm starten, wandelt das Grafikzeichen bei jedem Tastendruck Byte für Byte herunter. Wenn Sie jetzt noch einfügen:

```
7 POKE 53265,PEEK(53265)AND
247
```

das war ja das Einführen des 24-Zeilen-Modus, dann ist das Grafikzeichen zu Anfang fast versteckt. Wir sollten dann aber auch noch den Normalzustand wiederherstellen am Programmende:

```
60 POKE 53265,PEEK(53265)OR8
```

Auf diese Weise geschieht das fließende Scrollen nach unten. Verändern wir Zeile 10:

```
10 FOR I = 7 TO 0 STEP -1
```

dann erfolgt Scrollen nach oben.

Das gleiche Programm können wir mit kleinen Änderungen für horizontales Scrollen verwenden. Wir müssen nur überall dort, wo 53265 steht, jetzt 53270 und in Zeile 50 anstelle von OR3 jetzt OR0 einsetzen. Wenn Sie Zeile 10 mit der Rückwärtsschleife belassen haben, scrollt der Bildschirminhalt nach links. Ändern Sie dagegen Zeile 10 wieder zur ursprünglichen Vorwärtsschleife, dann erfolgt ein Scrollen nach rechts. Sie werden jetzt sagen, daß das alles ja ganz nett ist, aber was kann man damit tun? Damit kann man Laufschriften erzeugen, die in beliebiger Richtung über den Bildschirm sausen oder schleichen — je nachdem. Leider, leider kommen wir hier langsam aber sicher an die Grenzen von Basic. Denn jetzt taucht ein Problem auf, dessen Lösung — in Basic als auch in Maschinensprache — gleich ein zweites nach sich zieht. Sehen wir uns zunächst Problem Nummer 1 an:

In dem Moment, wo man das Zeichen geduldig Byte für Byte aus dem Versteck herausgeholt hat, also 7 in die Bits 0 bis 2 von 53265 oder 53270 eingegeben hat, muß man ja dafür sorgen, daß

1) der Bildschirminhalt um ein Zeichen weitergeschoben wird,

2) im Versteck (am Rand) ein nächstes Zeichen parat liegt.

Dazu können wir uns zum Beispiel des ohnehin schon eingebauten Zeilen-Scrolls bedienen, wie im nachfolgenden Programm:

```
10 POKE 53265,PEEK(53265)AND
247
20 PRINTCHR$(147):FOR I=1 TO
24:PRINTCHR$(17);:NEXT I
30 POKE 53265,(PEEK(53265)AND
248)OR7
40 PRINT"SCROLLING NACH
OBEN"
50 FOR I=6 TO 0 STEP -1
60 POKE53265,(PEEK(53265)AND-
248)OR I:FOR J=1TO 50:NEXT J
70 NEXT I:GOTO 30
```

Wenn Sie dieses Programm gestartet haben, können Sie mit RUN/STOP und RESTORE den Normalzustand wiederherstellen. Haben Sie etwas bemerkt? Ein Flackern des Bildes trat auf. Das ist das zweite Problem, auf das wir gleich kommen werden.

Ein Beispiel für das horizontale Scrollen gibt das Programm von H. Gehrmann in Happy-Computer, Ausgabe 4/84, Seite 108f. H. Gehrmann hat es geschafft, in Basic tatsächlich fließendes Scrollen ohne Bildschirmflackern zu erzeugen. In Zeile 36 seines Programms ist ein Teil seiner Lösung für Problem Nummer 2 zu finden. Dort steht:

```
36 IF PEEK(53265) <> 155 THEN
36
```

Erst danach druckt er an den oberen Bildschirmrand die zu scrollende Textzeile und scrollt dann, wie wir es bisher kennengelernt haben. Was ist denn nun dieses zweite Problem? Sehen wir dazu zunächst einmal in die Speicherstelle 53265 hinein. Geben Sie (nach NEW) mal die folgende Textzeile ein:

```
10 PRINT PEEK(53265):GOTO10
```

Nach RUN sehen Sie eine Reihe aus 27 und ab und zu taucht 155 auf. Wenn Sie jetzt mit RUN/STOP und RESTORE dieses Testprogramm angehalten haben, sehen wir uns diese Zahlen mal binär an:

27 = binär 0001 1011

155 = binär 1001 1011

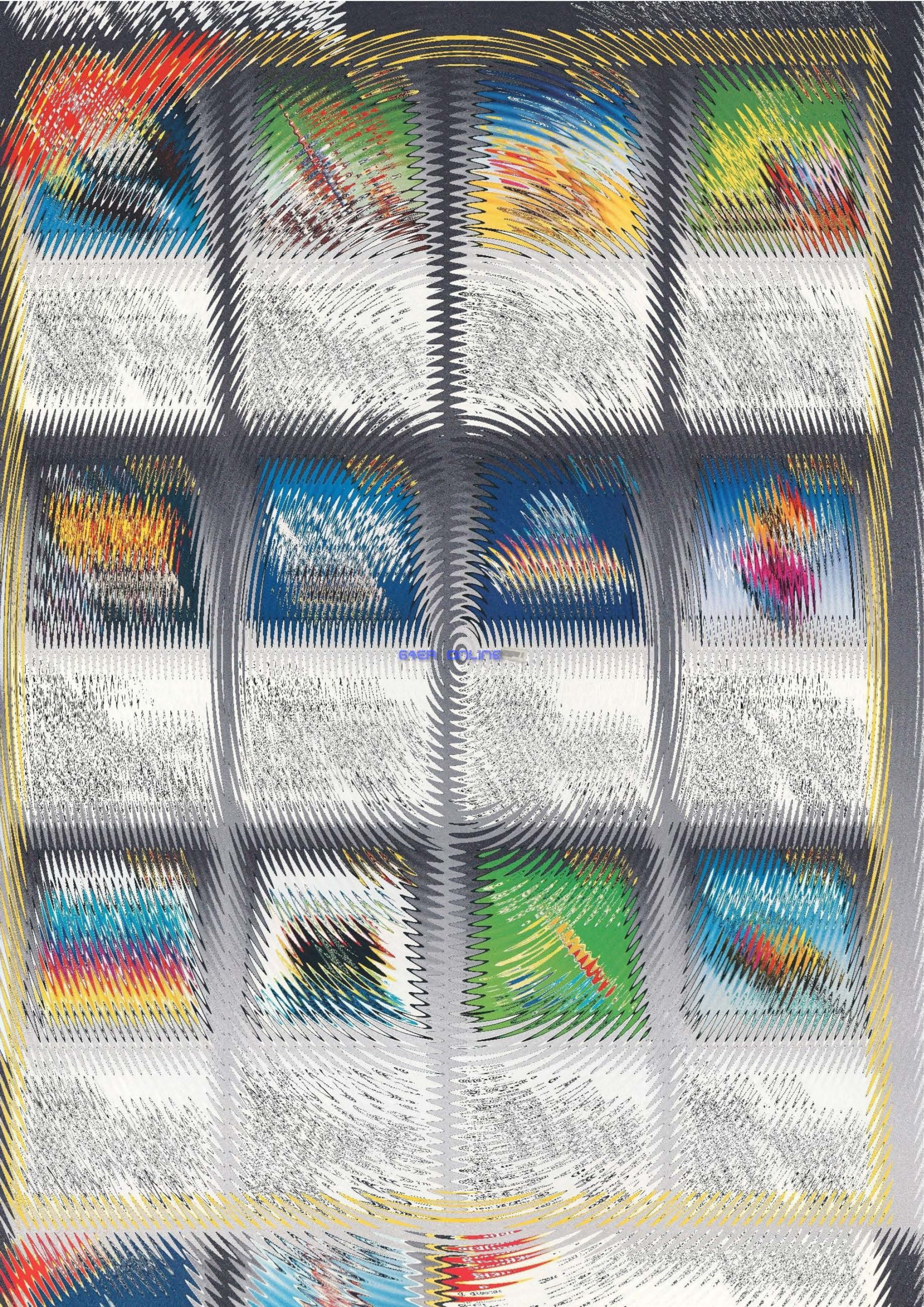
Alle Bits kennen wir schon, außer Bit 7, welches sich verändert hat. Es handelt sich um das msb des Rasterzeilen-Registers. Das LSB davon findet sich in Speicherzelle 53266. Wir haben es hier mit einer 9stelligen Binärzahl zu tun, von der 8 Stellen (die Bits 0 bis 7) in 53266 und die 9. Stelle als Bit 7 in 53265 zu finden sind. Gezählt werden ständig in diesen neun Bits die Rasterzeilen. Sie werden sich fragen, was das soll, hier ist die Antwort:

An der Entstehung eines Fernsehbildes ist ein Elektronenstrahl beteiligt. Er tastet Punktzeile für Punktzeile von oben nach unten den Bildschirm ab und erzeugt dabei das Bild. 20 mal in der Sekunde baut er ein neues Bild auf, und in welcher Punktzeile (Rasterzeile) er gerade ist, das zählt der Computer mit in diesem Rasterzeilenregister. Er muß deshalb mitzählen, weil er dem Elektronenstrahl einige Informationen übergeben muß, nämlich ob ein Punkt sichtbar sein soll oder nicht. Der Zeilenstrahl überstreicht den gesamten Bildschirm, also auch Bereiche, die außerhalb des lesbaren Textfensters liegen. Deswegen müssen wir auch weiter als bis 255 Rasterzeilen zählen. Weil aber in ein Byte (hier nun 53266) nur bis 255 gezählt werden kann, kommt noch das Bit 7 von 53265 als msb dazu.

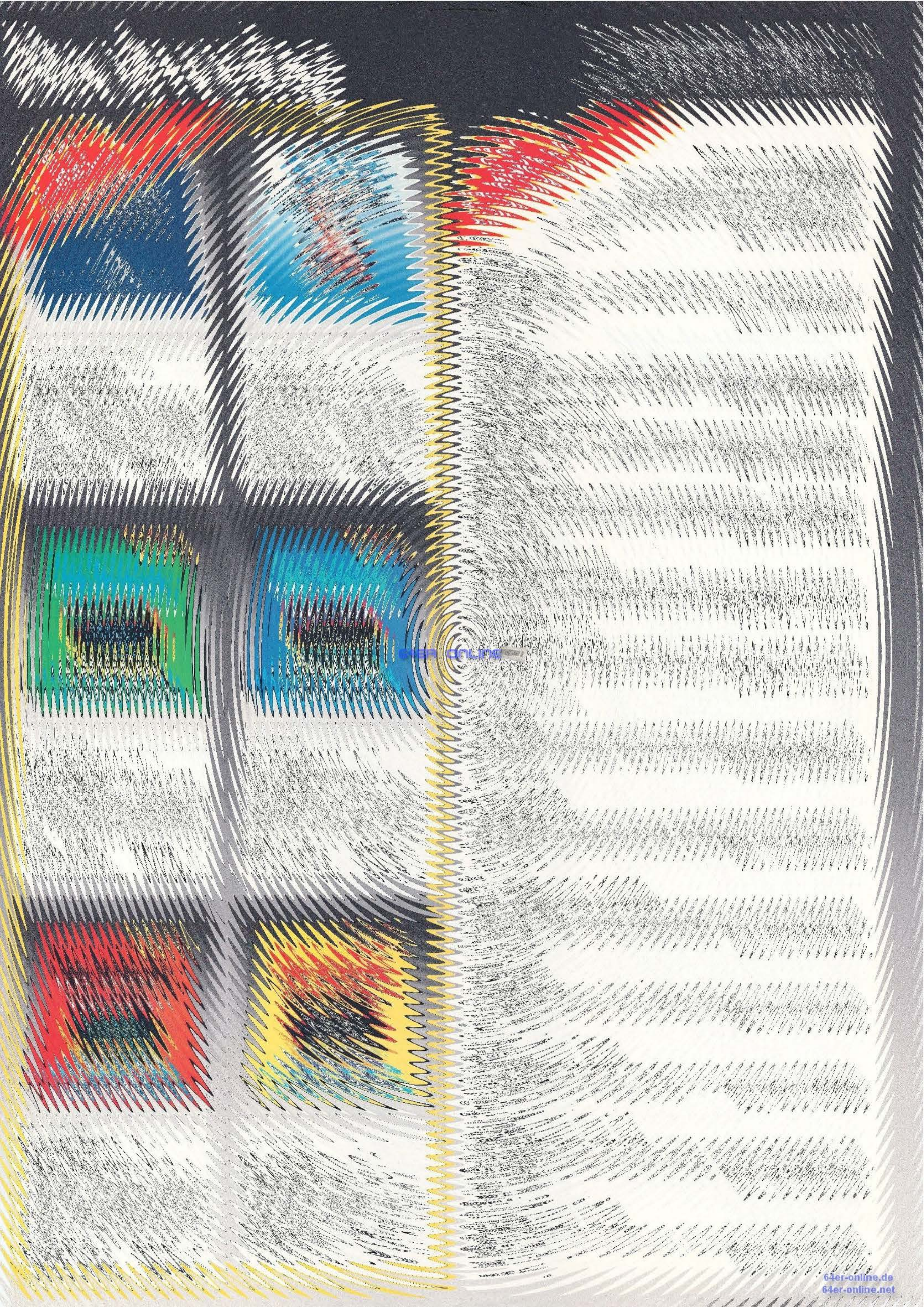
Jedesmal, wenn der Zeilenstrahl über die 255. Rasterzeile hinaushuscht, wird dieses Bit gleich 1. Das Problem Nummer zwei läßt sich also jetzt so ausdrücken: Wenn eine Bildänderung stattfindet, während der Zeilenstrahl über die Änderungsstelle hinwegstreicht, kommt es zu Zuckungen des Bildes. Man muß dafür sorgen, daß man diesem Flitzer nicht ins Gehege kommt. H. Gehrmann hat das in seinem Programm so gemanagt, daß er den Bildinhalt verändert, während der Rasterstrahl außerhalb des sichtbaren Bereiches arbeitet. Trotzdem gibt es hier aber noch Schwierigkeiten. Ein ganzer Bildschirm besteht aus 280 Rasterzeilen. Wenn man nun nachrechnet, braucht der Elektronenstrahl zirka 180 Microsekunden zum Überstreichen einer Rasterzeile, das heißt, er ist in sehr kurzer Zeit wieder im sichtbaren Bereich! Bis dahin muß die Bildveränderung abgeschlossen sein. Normalerweise

Fortsetzung auf Seite 150




64ER ONLINE


64er-online.de
64er-online.net


64er-online.de
64er-online.net

Fortsetzung von Seite 145

ist das eine Aufgabe für Maschinenprogramme, von denen Sie einige in der nachfolgend genannten Literatur finden:

— R. Babel, M. Krause, A. Dripke: Das Interface Age Systemhandbuch zum Commodore 64, München 1983, S. 63-64,

— A. Plenge: Das Grafikbuch zum Commodore 64, Düsseldorf 1984, S. 272ff.

Dornröschen ist wieder dran: 2D-Grafik besser

Von nun an kümmern wir uns wieder um Dornröschen, die hochauflösende Grafik. Wie Sie wohl spätestens in Folge 4 bemerkt haben, ist das Koordinatensystem, das unser Commodore 64 auf dem Bildschirm einrichtet, reichlich unüblich. Zur Erinnerung sehen Sie sich mal Bild 1 an.

Wie Sie aus der Schule vielleicht noch wissen, ist die normale Darstellung eines Koordinatensystems aber so wie in Bild 2 gezeigt.

Im Listing 3 der 4. Folge mußte deshalb eine einfache Sinus-Funktion so verändert werden, daß man sie von der Formel her kaum mehr erkennen kann (Zeile 5). Bleiben wir beim Beispiel der Sinus-Kurve. Die sieht so aus wie in Bild 3 gezeigt.

Wie groß oder klein X also auch immer wird, Y bewegt sich nur zwischen +1 und -1 hin und her. Wenn wir aber die normale Sinus-Gleichung (Y=SIN(X)) auf unserem Bildschirmsystem zeichnen lassen, werden wir gar nichts oder kaum etwas sehen (Bild 4).

Wir müssen also einen Weg finden, ein uns genehmes Koordinatensystem anstelle des Bildschirmsystems zu verwenden. Dazu soll uns die Mathematik helfen. Man nennt das, was wir jetzt tun werden, eine Transformation! Keine Angst, es wird nicht zu schwierig!

In Bild 5 sind zwei Koordinatensysteme abgebildet.

1) Das Bildschirmsystem X, Y mit Nullpunkt O

2) Das von uns gewünschte System X', Y', mit Nullpunkt O', welches rot gezeichnet ist.

Außerdem ist der Punkt P1 zu sehen. Je nach dem, auf welches Koordinatensystem er bezogen wird, hat er die Koordinaten

1) X1, Y1 oder

2) X1', Y1'.

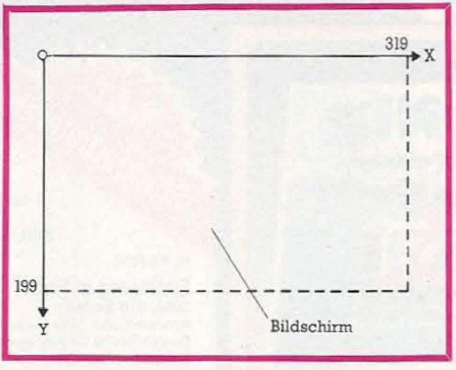


Bild 1. Das Bildschirmkoordinatensystem

Unser eigenes System soll vom unteren Wert XU bis zum oberen X-Wert XO reichen und vom unteren Y-Wert YU bis zum oberen YO. Am Beispiel unserer Sinus-Kurve wäre es sinnvoll, die X-Werte von XU=-1 bis XO=10 und die Y-Werte von YU=-2 bis YO=+2 abzubilden. Die Entfernung von XU bis XO (also XO-XU) erstreckt sich auf 319 Bildpunkte. Man kann nun ein Verhältnis bilden, denn X1 muß sich zu (X1'-XU)(das ist die Länge der Strecke von XU bis X1') genauso verhalten wie 319 zur Gesamtentfernung (XO-XU). Als Formel sieht das so aus:

$$\frac{X1}{X1' - XU} = \frac{319}{XO - XU}$$

Das einzige in dieser Formel, was wir nicht kennen (X1' ist ja unsere Eingabe, die auf unser eigenes Koordinatensystem bezogen ist) ist nun X1, also die X-Koordinate unseres Punktes P1 im Bildschirmsystem. Man kann diese Verhältnisgleichung nach X1 auflösen:
X1=(X1' - XU)/(XO - XU)*319

So ähnlich geht das auch bei der Y-Koordinate. Es ist nur zu beachten, daß im Bildschirmsystem die Y-Achse nach unten, in unserem System aber nach oben zeigt. Deswegen ist die Entfernung von YO nach Y1': YO-Y1'. Das Verhältnis hat hier also die Formel:

$$\frac{Y1}{YO - Y1'} = \frac{199}{YO - YU}$$

Nach Y1 aufgelöst ergibt sich:
Y1=(YO-Y1')/(YO-YU)*199

Es empfiehlt sich, diese beiden sogenannten Transformationsgleichungen als Basicfunktionen zu definieren:

```
DEFFNX(X)=(X-XU)/(XO-XU)*319
DEFFNY(Y)=(YO-Y)/(YO-YU)*199
```

Nun können wir ein kleines Aufrufprogramm schreiben, welches sich wieder der Unterprogrammsammlung aus Folge vier bedient. Laden Sie also zuerst (die hoffentlich damals gespeicherten!) Unterprogramme und tippen Sie dann ein:

```
5 POKE 52,92:POKE 56,92
100 DEFFNX(X)=(X-XU)/(XO-XU)
*319
110 DEFFNY(Y)=(YO-Y)/)YO-YU)
*199
120 REM +++++ BEISPIELFUNK-
TION +++++
140 DEFFNA(X)=SIN(X)
150 PRINTCHR$(147)CHR$(17):
INPUT"XU,XO,YU,YO=";XU,XO,
YU,Y0
160 INPUT"ZEICHEN- UND
HINTERGRUNDFARBE=";F1,F2
170 GOSUB 50100:FOR I=XU TO XO
STEP(XO-XU)/319
180 X=FNX(I):Y=FNY(FNA(I)):GO-
SUB 50040
190 NEXT I:X1=FNX(XU):Y1=FNY
(O):X2=FNX(XO):Y2=Y1
GOSBUB 50060
```

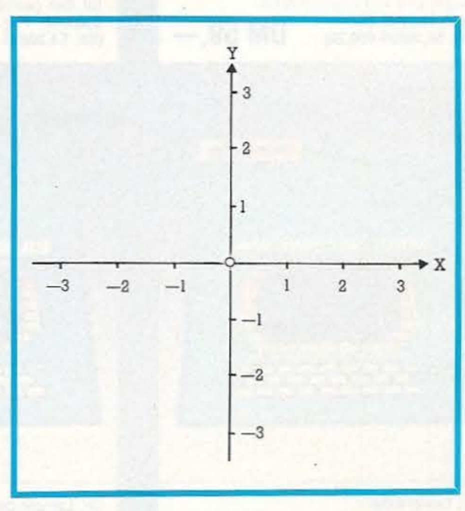


Bild 2. Das gewohnte Koordinatensystem

```
195 X1=FNX(O):Y1=FNY(YU):X2=
X1:Y2=FNY(YO):GOSUB 50060
200 GET A$:IF A$="" THEN 200
210 GOSUB 50030
220 END
```

Nach RUN und mit der erforderlichen Geduld (kennen Sie ja schon) zeichnet der Computer eine ganz normale Sinuskurve in das von Ihnen durch XU,XO,YU und YO angegebene Koordinatensystem. In die Zeile 140 können Sie jetzt jede beliebige Funktion eingeben. Allerdings ist es unglücklich, daß man jedesmal das Programm ändern muß, nämlich Zeile 140, wenn man eine neue Funktion sehen will. Es gibt da einen Trick, der mit dem Tastaturpuffer arbeitet (Speicherstellen 631-640) und den ich Ihnen nun zeigen möchte. Zunächst fügen wir noch eine Zeile 130 ein:

```
130 F$="SIN(X)"
```

Wenn das Programm startet, soll man zunächst einmal erfahren, welche Funktion überhaupt in Zeile 140 steht. Deswegen also:

```
10 PRINTCHR$(147):POKE211,5:
```

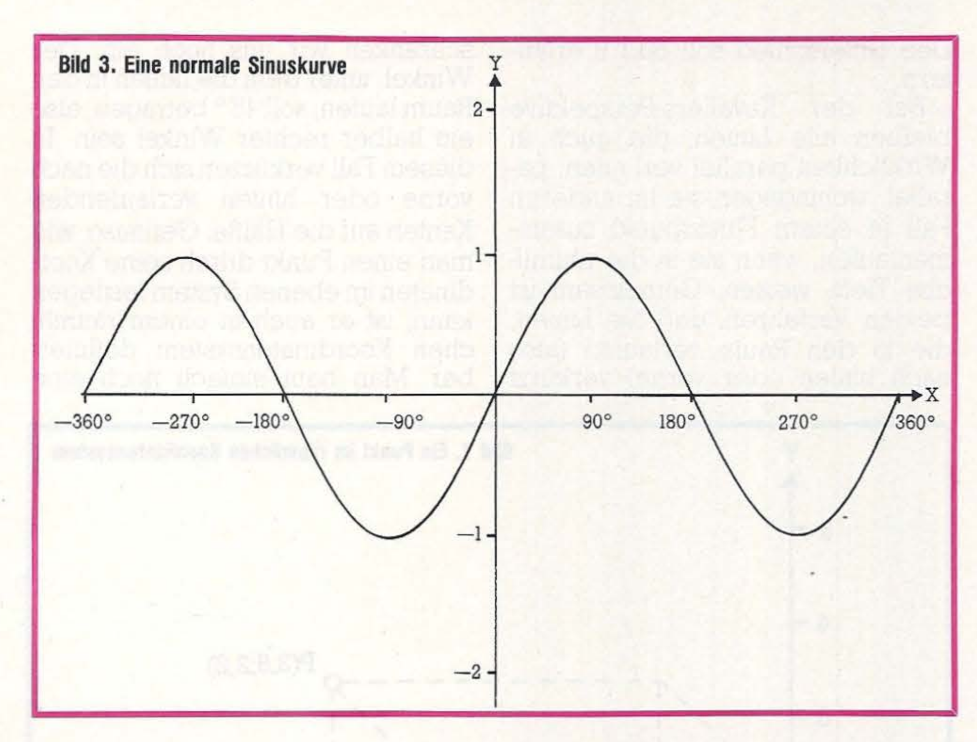


Bild 3. Eine normale Sinuskurve

POKE214,10:SYS58640:
PRINT"FUNKTION IM PROGRAMM:"
20 PRINT:PRINTTAB(3)"Y="F$

Wenn wir das so laufen lassen, wird keine Funktion erscheinen, weil diese dem Computer erst in Zeile 130 mitgeteilt wird. Der Computer muß also vorher nachsehen, als was F$ definiert ist. Deswegen führen wir ein:

15 K=1:GOSUB 130:K=0

und

145 IF K=1 THEN RETURN

Jetzt druckt das Programm uns die Funktion aus. Als nächstes soll der Computer erfahren, ob die Funktion zu verändern ist:

30 PRINT:PRINTTAB(5)CHR$(18)"A" CHR$(146)"LTE ODER" CHR$(18)"N"CHR$(146)"EUE FUNKTION? "
35 GET A$:IFA$<>"A" AND A$<> "N" THEN 35

Wenn die alte Funktion gewählt wurde, dann kann alles wie bisher ablaufen:

40 IF A$="A" THEN 100

Ansonsten wollen wir jetzt unseren Trick anwenden. Zunächst das Eingeben der neuen Funktion:

45 PRINT:PRINTTAB (3);:INPUT "Y=";F$

Dann:

50 PRINTCHR$(147)CHR$(17)CHR$ (17)"130F$="CHR$(34)FCHR(34)
55 PRINT"140DEFFNA(X)="F$
60 PRINT"RUN 100":PRINTCHR$ (19);
65 POKE 631,13:POKE 632,13:POKE 633,13:POKE 198,3:END

Tippen Sie's ein und fügen Sie in Zeile 100 noch ein STOP ein, damit Sie sehen können, was mit diesen

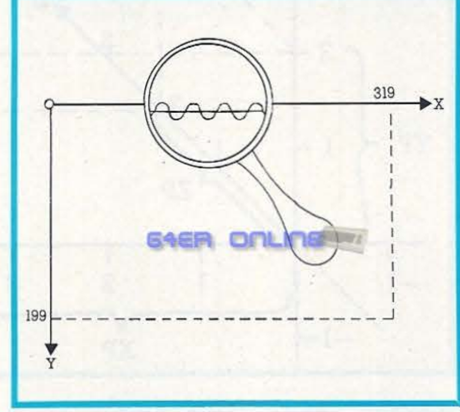


Bild 4. Die normale Sinuskurve im Bildschirmkoordinatensystem

Zeilen passiert. Sie sehen, daß der Bildschirm zunächst frei gemacht wird (CHR$(147) in Zeile 50) und dann zwei Programmzeilen 130 und 140 gedruckt werden mit der neuen Funktion, danach noch die RUN-Anweisung. Der Cursor springt dann wieder in die Ecke links oben (CHR$(19) in Zeile 60). Jetzt arbeitet unser Computer den Inhalt des Tastaturpuffers ab, der aus drei RETURNs (die 13) besteht. Die Anzahl 3 muß in Speicherstelle 198 festgehalten sein. Die Tatsache, daß der Tastaturpuffer abgearbeitet wird, haben wir dem END zu verdanken. Weil aber ein RETURN nach RUN 100 folgt, startet das Programm sofort wieder selbständig ab Zeile 100. Wenn Sie jetzt mal Zeile 130 und 140 listen lassen, steht dort die neue Funktion. Weil sie sich im Programmablauf störend auswirken, sollen die zu druckenden Zeilen unsichtbar werden. Deswegen fügen wir in Zeile 45 noch an:

:POKE 646,6

und ab Zeile 100 soll wieder sichtbar werden, was zu drucken ist, weshalb wir dort anfügen:

:PRINTCHR$(147):POKE 646,14

Dieser Trick ist sehr vielseitig anwendbar. Man muß nur beachten, daß durch das RUN alle Variablen, die bis dahin definiert wurden, gelöscht werden. Das ist ja auch sinnvoll, weil wir das Basicprogramm verändert haben und so unter Umständen Variable überschrieben wurden. Sollte es nicht möglich sein, alle Variablen nach dem Neustart zu definieren, dann läßt sich ein ähnlicher Kniff anwenden, wie wir ihn in Zeile 15 verwendet haben. Probieren Sie das Programm mal mit folgenden Eingaben:

Y=SIN(X) mit XU=-1,XO=10,YU=-2,YO=2
oder Y=EXP(-X/10)*COS(X) mit XU=-18,XO=10,YU=-5,YO=5
oder Y=2*SIN(X)+SIN(2*X+2) mit XU=-6,XO=15,YU=-3,YO=3

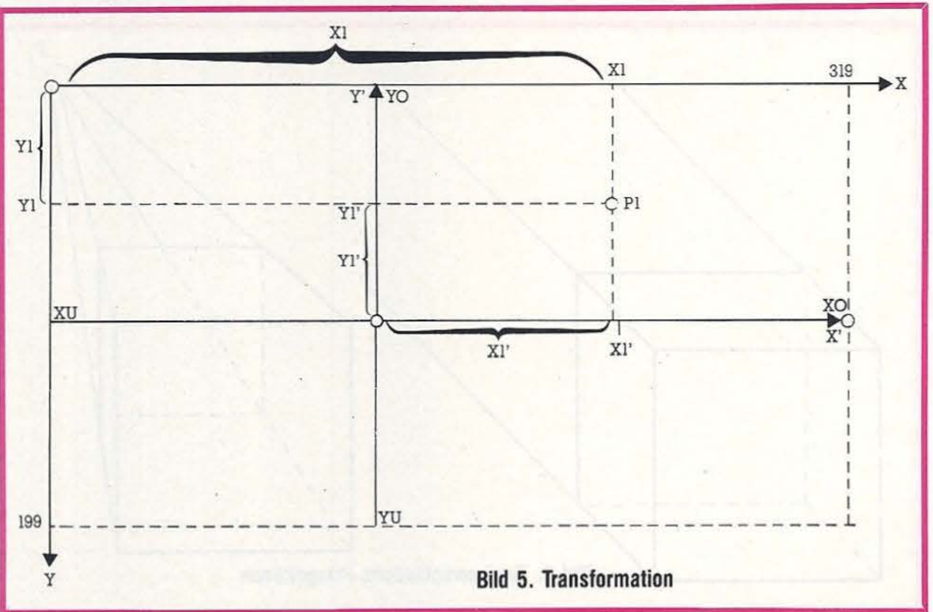


Bild 5. Transformation

Ihrer Phantasie sind keine Grenzen gesetzt.

Flacher Raum: 3D-Grafik, wie funktioniert das?

Nun sind wir bei dreidimensionalen, also räumlichen Abbildungen angekommen. Zunächst einmal: Es ist klar, weil der Fernsehschirm eben nur eine Fläche ist wie ein Blatt Zeichenpapier, daß wir etwas Räumliches auf einer Fläche darstellen müssen. Alles andere ist Science fiction, auch für die größten Computer! Bei 3D-Darstellungen unterscheidet man hauptsächlich zwei Arten:

a) Zeichnungen, die mit einer Spezialbrille räumlich wahrgenommen werden können,

b) Zeichnungen, die perspektivisch gestaltet sind und deswegen als räumlich empfunden werden.

Im Prinzip ist der Weg a) für uns begehbar. Wir könnten die dazu nötigen zwei verschiedenfarbigen Zeichnungen (meist rot und blau) mit Hilfe des Mehrfarben-Bitmap-Modus erzeugen. Allerdings ist der rechnerische Aufwand nicht unerheblich. Ab und zu werden in Programmzeitschriften entsprechende Entwürfe veröffentlicht.

Der übliche Weg allerdings geht über eine perspektivische Darstellung. Zwei Arten wiederum finden vor allem Anwendung:

a) Die sogenannte Kavaliers-Perspektive

b) Die Perspektive mit mindestens einem Fluchtpunkt.

Den Unterschied soll Bild 6 erläutern.

Bei der Kavaliers-Perspektive bleiben alle Linien, die auch in Wirklichkeit parallel verlaufen, parallel, wohingegen sie im anderen Fall in einem Fluchtpunkt zusammenlaufen, wenn sie in die räumliche Tiefe weisen. Gemeinsam ist beiden Verfahren, daß die Linien, die in den Raum verlaufen (also nach hinten oder vorne) verkürzt werden, weil unsere Augen sonst über die Größenverhältnisse täuschen. Zwar sieht die Fluchtpunktperspektive sehr viel natürlicher aus, aber die Kavaliers-Perspektive ist mathematisch bei weitem nicht so kompliziert. Deswegen wollen wir uns mit ihr befassen. Weiterhin schränken wir uns noch ein: Der Winkel, unter dem die Linien in den Raum laufen, soll 45° betragen, also ein halber rechter Winkel sein. In diesem Fall verkürzen sich die nach vorne oder hinten verlaufenden Kanten auf die Hälfte. Genauso, wie man einen Punkt durch seine Koordinaten im ebenen System festlegen kann, ist er auch in einem räumlichen Koordinatensystem definierbar. Man baut einfach noch eine dritte Achse (eine in den Raum weisende Z-Achse) zu den vorhandenen dazu und erhält so Zusammenhänge wie in Bild 7.

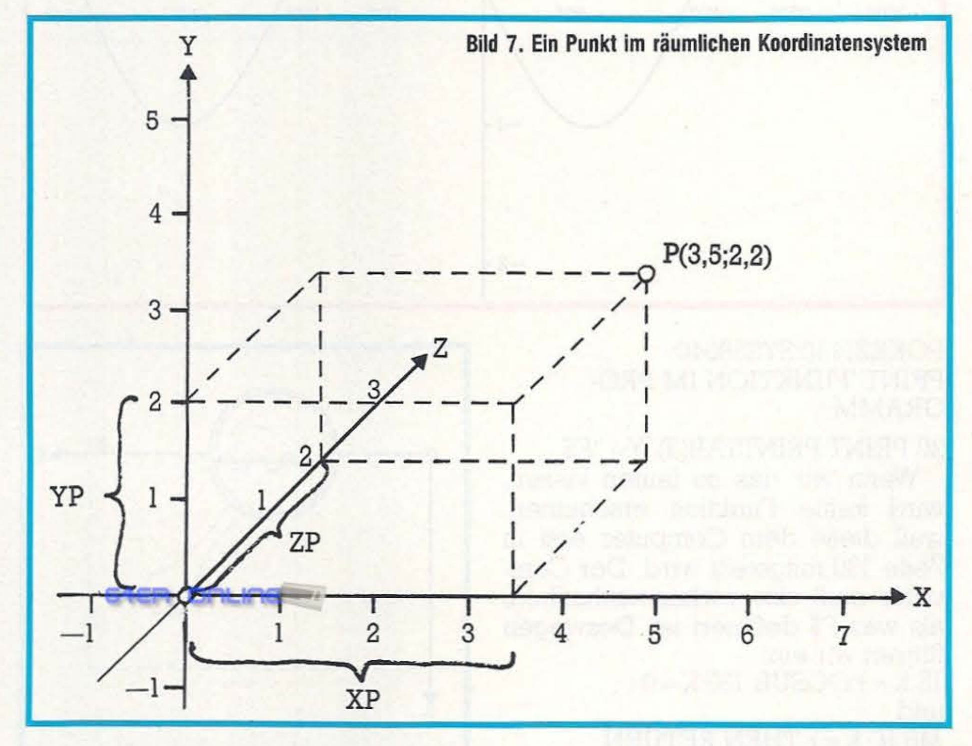


Bild 7. Ein Punkt im räumlichen Koordinatensystem

Das Verfahren soll durch diese Erklärung noch etwas verdeutlicht werden. Man kann sich den Raum als eine große Anzahl dicht an dicht hintereinander gestapelter X-Y-Ebenen vorstellen, etwa wie in Bild 8.

Im Bild sind sie, damit man das erkennen kann, mit Lücken gezeichnet.

Auf einer Fläche zeichenbar wird das Verfahren dann, wenn man jede X-Y-Ebene um den Betrag ihrer Raumtiefe (das entspricht der halben Z-Koordinate) entlang der Achse Z nach rechts oben verschiebt wie in Bild 9.

Nehmen wir nun noch an, daß auf jeder Ebene eine Kurve gezeichnet sei, die sich von Ebene zu Ebene etwas verändert, dann bekommen wir auf diese Weise aus allen Kurven zusammen den Eindruck einer räumlichen Fläche wie in Bild 10.

Das Prinzip ist jetzt wohl klar. Wenn wir nun bedenken, daß die

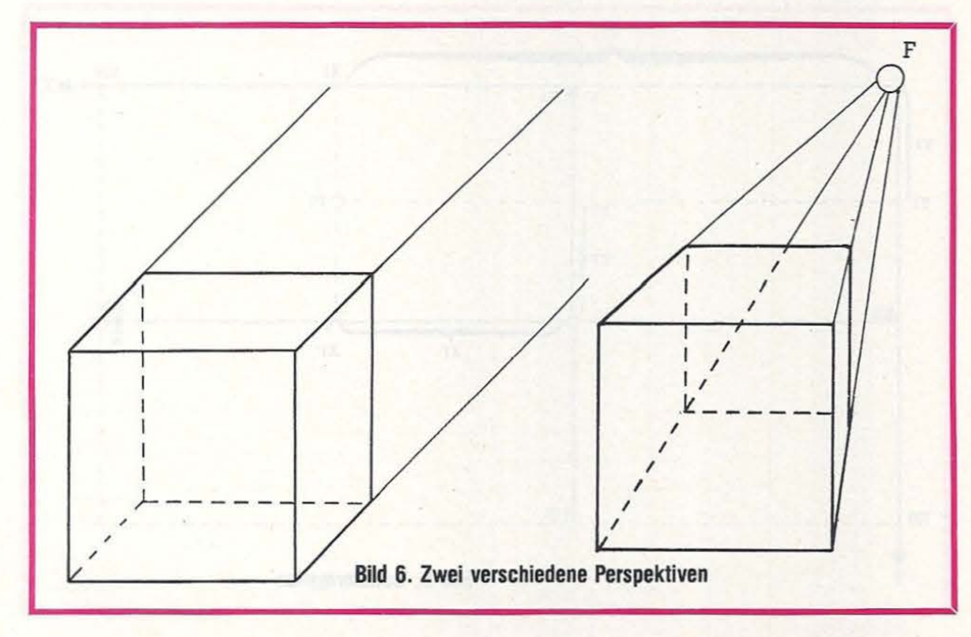


Bild 6. Zwei verschiedene Perspektiven

Raumfläche aus lauter ebenen Kurven und die Kurven aus lauter Punkten zusammengesetzt sind, dann stellt sich für uns die Frage, wie man einen Punkt P(X1',Y1',Z1') an der richtigen Stelle unseres Bildschirmes darstellen kann. Weil wir schon wissen (von den 2D-Zeichnungen her), wie wir unser gewünschtes Koordinatensystem auf den Bildschirm bringen können, stellt sich die Frage für uns einfacher. Wir müssen uns nur noch überlegen, wie man mit zwei Koordinaten X1,Y1 den räumlichen Punkt in unserem selbstgewählten System zeichnen kann. Sehen wir uns dazu Bild 11 an.

Wir haben hier einfach so getan, als gäbe es die Z-Achse gar nicht, sondern der Punkt P wäre einfach um einen Wert ΔX nach rechts und einen weiteren Wert ΔY nach oben geschoben worden. Die Koordinaten von P im ebenen System sind dann

X1 = X1' + ΔX
Y1 = Y1' + ΔY

wie man auch aus der Zeichnung sehen kann. Wie lang ΔX ist, kann man aus dem Dreieck ABC berechnen. Mathematisch Versierte werden mir zustimmen, daß

COS(α) = ΔX/Z1

ist und im Dreieck DEF gilt

SIN(α) = ΔY/Z1

Daraus ergibt sich dann:

ΔX = Z1*COS(α)
ΔY = Z1*SIN(α)

Weil außerdem noch nach unserer anfänglichen Vereinbarung α = 45° beträgt und die Z-Achse auf die Hälfte verkürzt ist und weil schließlich

$SIN(45°) = COS(45°) = 1/\sqrt{2}$ ist, ergibt sich:

ΔX = Z1'/(2*SQR(2))
ΔY = Z1'/(2*SQR(2))

Beide sind also gleich und für unser eigenes Koordinatensystem hat der Punkt P die Koordinaten:

X1 = X1' + Z1'/(2*SQR(2))
Y1 = Y1' + Z1'/(2*SQR(2))

Damit ist unser Problem gelöst. Denn X1', Y1' und Z1' sind die vorgegebenen Koordinaten und X1,Y1 können jetzt mit unseren Transformationen X = FNX(X1),Y = FNY(Y1) in Bildschirmkoordinaten umgerechnet werden. Wir verschachteln nun zwei Schleifen ineinander wie in Bild 12 und lassen uns damit Kurve für Kurve zeichnen.

Nun wollen wir auch für die 3D-Grafik ein Programm entwerfen. Tippen Sie also NEW ein, laden Sie die Grafik-Unterprogramme und erwecken Sie unser 3D-Dornröschen mit dem nachfolgenden Aufrufprogramm.

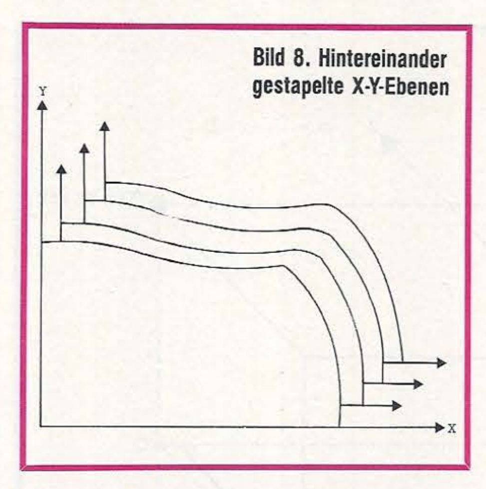


Bild 8. Hintereinander gestapelte X-Y-Ebenen

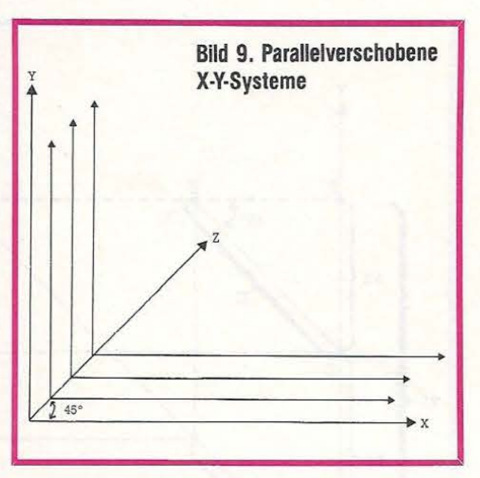
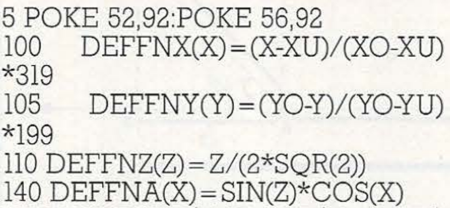


Bild 9. Parallelverschobene X-Y-Systeme



```
5 POKE 52,92:POKE 56,92
100     DEFFNX(X)=(X-XU)/(XO-XU)
*319
105     DEFFNY(Y)=(YO-Y)/(YO-YU)
*199
110 DEFFNZ(Z)=Z/(2*SQR(2))
140 DEFFNA(X)=SIN(Z)*COS(X)
150 PRINTCHR$(147)CHR$(17)CHR$
(18)"UNSER    SYSTEM:"CHR$(146):
PRINT:PRINT
160 INPUT"XU,XO=";XU,XO:INPUT
"YU,YO=";YU,YO
```

Jetzt wird es ein bißchen kompliziert, weil wir darauf achten müssen, daß die Z-Achse noch auf dem Bildschirm paßt. Wir lassen uns berechnen und anzeigen, wie groß ZO höchstens sein darf und wie klein ZU:

```
162 Z3=2*X0*SQR(2):Z4=2*YO*
SQR(2):IF Z3<Z4 THEN PRINT
"ZO MAXIMAL="Z/3:GOTO 166
164 PRINT"ZO MAXIMAL="Z4
166 Z5=2*XU*SQR(2):Z6=2*YU*
SQR(2):IF Z5>Z6 THEN PRINT "ZU
MINIMAL="Z5:GOTO170
168 PRINT"ZU MINIMAL="Z6
170 PRINT:INPUT"ZU,ZO=";ZU,ZO:
Z1=FNZ(ZO):Z2=FNZ(ZU)
```

Im nachfolgenden geht es nach dem (nicht immer angebrachten) Motto: Vertrauen ist gut, Kontrolle besser:

```
172 IF Z1>YO OR Z1>YO THEN 162
174 IF Z2<XU OR Z2<YU THEN 162
```

Dann lassen wir uns die Freiheit der Farbenwahl:

```
180 PRINT:INPUT"ZEICHEN- UND
HINTERGRUNDFARBE=";F1,F2
```

Nach dem Anschalten der Hochauflösung zeichnen wir die Achsen:

```
190 GOSUB 50100:X1=FNX(XU):Y1=
FNY(0):X2=FNX(XO):Y2=Y1:
GOSUB 50060
192 X1=FNX(O):Y1=FNY(YU):X2=
X1:Y2=FNY(YO):GOSUB 50060
194   X1=FNX(Z2):Y1=FNY(Z2):X2=
FNX(Z1):Y2=FNY(Z1):GOSUB 50060
```

Wir könnten jetzt mit den beiden ineinandergeschachtelten Schleifen beginnen, wenn es da nicht noch das Problem der Schrittweite in den Schleifen gäbe. Die Schrittweite von X haben wir schon im 2D-Programm verwendet:

```
200 DX=(XO-XU)/319
```

Die Schrittweite von Z ist nicht so einfach zu fassen, weil man ja, je

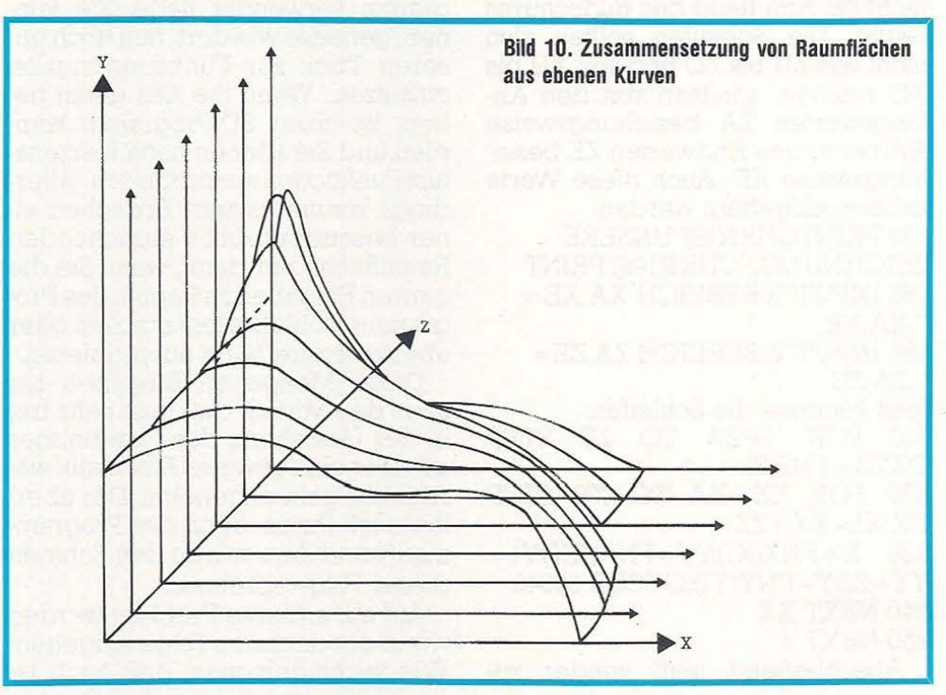


Bild 10. Zusammensetzung von Raumflächen aus ebenen Kurven

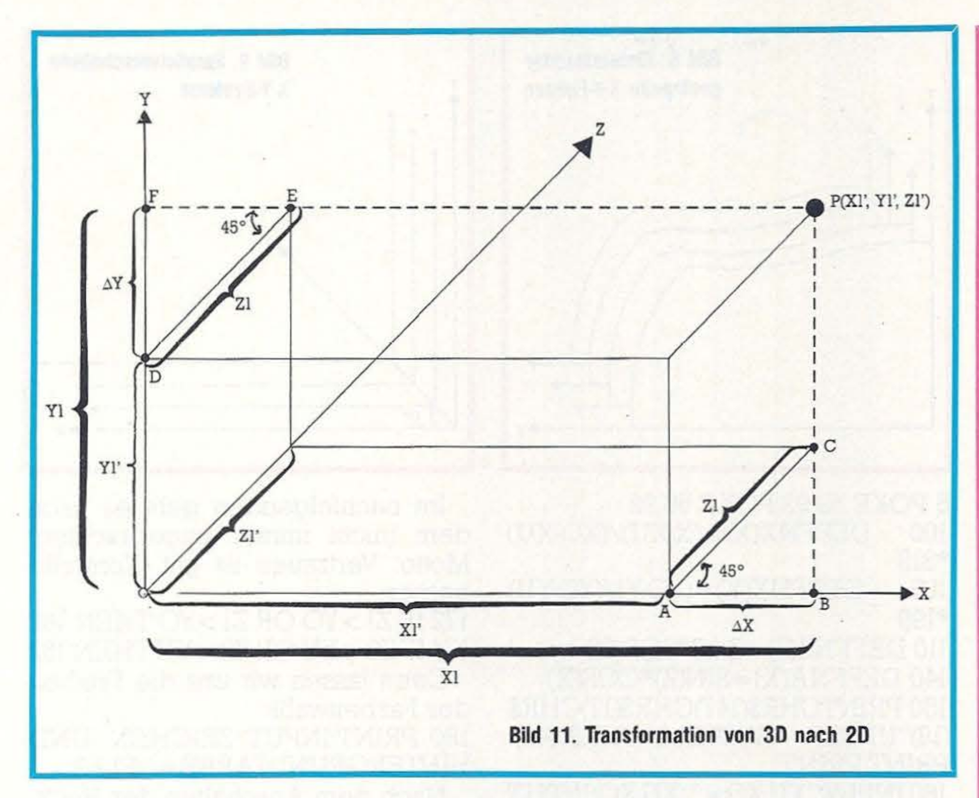


Bild 11. Transformation von 3D nach 2D

nach Kurvenart, mal weitere und mal engere Schritte machen muß. Deswegen halten wir diese Schrittweite veränderlich und lassen uns vorher nochmal danach fragen:

```
202 DZ=A*(Z1-Z2)/(FNX(Z1)-FNX
(Z2))
182 INPUT"SCHRITTWEITE VON
Z(CA.8)=";A:PRINT
```

Mit Zeile 202 hätten wir bei A=1 die selbe Punktauflösung wie in der X-Schleife. Das ist meistens zu eng und A sollte deshalb Werte um etwa 8 haben.

Es ist ganz gut, sich auch die Freiheit offenzulassen, von wo bis wo unsere Raumfläche zu zeichnen ist. Es sieht einfach besser aus, wenn sie nicht bis zum Rand des Bildschirms reicht. Die Schleifen sollten also nicht von ZU bis ZO und von XU bis XO reichen, sondern von den Anfangswerten ZA beziehungsweise XA bis zu den Endwerten ZE beziehungsweise XE. Auch diese Werte müssen abgefragt werden:

```
184 PRINTCHR$(18)"UNSERE
ZEICHNUNG:"CHR$(146):PRINT
185 INPUT"X-BEREICH XA,XE=
";XA,XE
186 INPUT"Z-BEREICH ZA,ZE=
";ZA,ZE
```

Jetzt kommen die Schleifen:

```
210 FOR Z=ZA TO ZE STEP
DZ:ZZ=FNZ(Z)
220 FOR XX=XA TO XE STEP
DX:X1=XX+ZZ
230 X=FNX(X1):YY=FNA(XX):Y1=
YY+ZZ:Y=FNY(Y1):GOSUB 50040
240 NEXT XX
250 NEXT Z
```

Abschließend muß wieder zurückgeschaltet werden auf den Normalbildschirm:

```
300 GET A$:IF A$="" THEN 300
310 GOSUB 50030
320 END
```

So! Nun können Sie das ganze vorsichtshalber abspeichern und dann mit RUN starten. Aber fassen Sie sich in Geduld. Bis das Bild komplett ist, vergehen zirka 21 Minuten. Probieren Sie mal die Eingaben: XU=-1,XO=6,YU=-2,YO=6,ZU=-1, ZO=6, Schrittweite von Z=8, XA=0, XE=4,ZA=0,ZE=4.

Es wird Ihnen sicherlich aufgefallen sein, daß ich zu Beginn des Programms den gleichen Zeilennummernabstand wie beim 2D-Programm verwendet habe. Sie können, genauso wie dort, nun noch unseren Trick zur Funktionseingabe einsetzen. Wenn Sie das getan haben, ist unser 3D-Programm komplett und Sie können nach Herzenslust Funktionen ausprobieren. Allerdings kommt es zum Erreichen einer besonders schön aussehenden Raumfläche nur dann, wenn Sie die ganzen Eingaben zu Beginn des Programms wohlüberlegt machen oder aber mehrere Male ausprobieren.

Diese Menge an Eingaben hat zwar den Vorteil, daß man sehr frei in der Gestaltung des Ergebnisses ist, aber eine gewisse Automatik wäre schon ganz angenehm. Das überlasse ich Ihnen, denn der Programmaufwand dazu würde den Rahmen dieser Folge sprengen.

Auf ein anderes Problem werden wir in der nächsten Folge eingehen: Wie verhindert man, daß auch Linien gezeichnet werden, die eigentlich beim realen Körper (oder bei der Fläche) nicht sichtbar sind, weil sie hinter anderen Teilen der Zeichnung verschwinden?

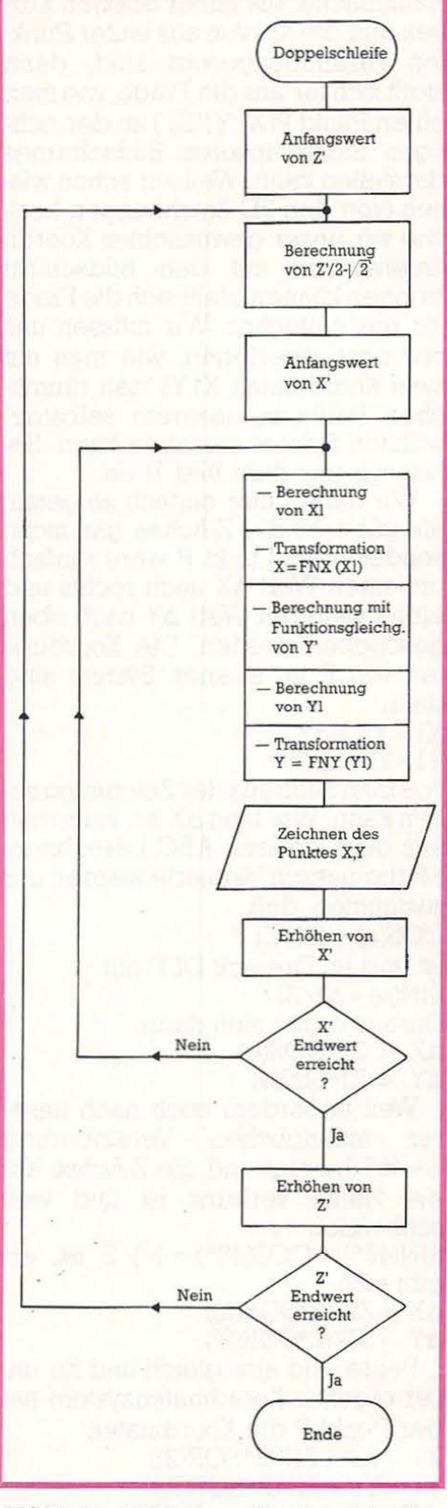


Bild 12. Das Flußdiagramm für die verschachtelten Schleifen

Ich hoffe, daß diese Folge nicht allzu schwierig war. Sollten Sie Probleme mit dem mathematischen Teil haben, trösten Sie sich, indem Sie die Programme einfach so abtippen. Oder stellen Sie Fragen per Leserbrief. Ich werde mich bemühen, alle zu beantworten.

(Heimo Ponnath/aa)

Der gläserne VC 20



Der VC 20, schon etwas betagt und oft genug totgesagt, ist immer noch der meistverbreitete Computer seiner Klasse.

Mit diesem Kurs wollen wir den legendären »Volkscomputer« endlich für alle Anwender vollkommen transparent machen.

Das Betriebssystem und das Basic des VC 20 sind äußerst flexibel gestaltet. Es gibt viele Möglichkeiten, das Bestehende zu ändern oder zu ergänzen. Diese Serie soll über die üblichen Tips und Tricks hinausgehen. Es werden also nicht nur POKEs, sondern auch weitergehende Maschinenprogramme wie zum Beispiel Funktionstastenabfrage oder die Definition neuer Basic-Befehle besprochen. Dieser erste Teil soll dabei bereits einen tieferen Einblick in das VC 20-System geben.

Wie Basic den Speicher verwaltet

Beginnen wir mit der Organisation des verfügbaren RAM durch den Basic-Interpreter.

Der Basicbeginn liegt bei Adresse 4096, das Ende bei Adresse 7680 (die Werte beziehen sich auf die Grundversion). Unmittelbar ab dem Basicbeginn wird das eigentliche Programm abgelegt. An dessen Ende beginnen die Variablen und Felder (Bild 1).

Der Variablenbereich wächst beim Anlegen neuer Variablen von unten nach oben. Nur das Stringende wandert in entgegenlaufender Richtung.

Die wichtigsten Zeiger, wie unter anderem Beginn und Ende von Basic und Variablen, sind in der Zeropage (Adresse 0 bis 256) abgelegt (Tabelle 1). Dabei ist die Reihenfolge Low-Byte/High-Byte zu beachten (Adresse = High-Byte * 256 + Low-Byte).

Um Speicherplatz für Maschinenprogramme oder Sonderzeichen zu schaffen, hat man prinzipiell zwei Möglichkeiten. Entweder man verschiebt den Basicanfang nach oben oder das Basicende nach unten. Letztere Alternative ist in den meisten Fällen günstiger.

Um zum Beispiel das Basicende von Adresse 7680 nach 7168 (= 512 Byte) zu verlegen, gibt man ein:
POKE 55,0:POKE 56,28:CLR:REM (256 * 28 = 7168)

Bei anderen Speichergrößen verfährt man analog.

Der Befehl CLR ist nötig, damit sich verschiedene Hilfszeiger (Stringbeginn und Felderende) anpassen können.

Tabelle 1. Der Inhalt dieser Speicherstellen bestimmt die Aufteilung des Basic-RAM

Speicherstelle	Bezeichnung	Werte
43 44	BASIC BEGINN	GV : 4096 + 3K : 1024 + 8K : 4608
45 46	VARIABLEN BEGINN	Abhängig von der Programmlänge
55 56	BASIC ENDE	GV : 7680 + 3K : 7680 + 8K : 16384 + 16K : 24576 + 24K : 32768

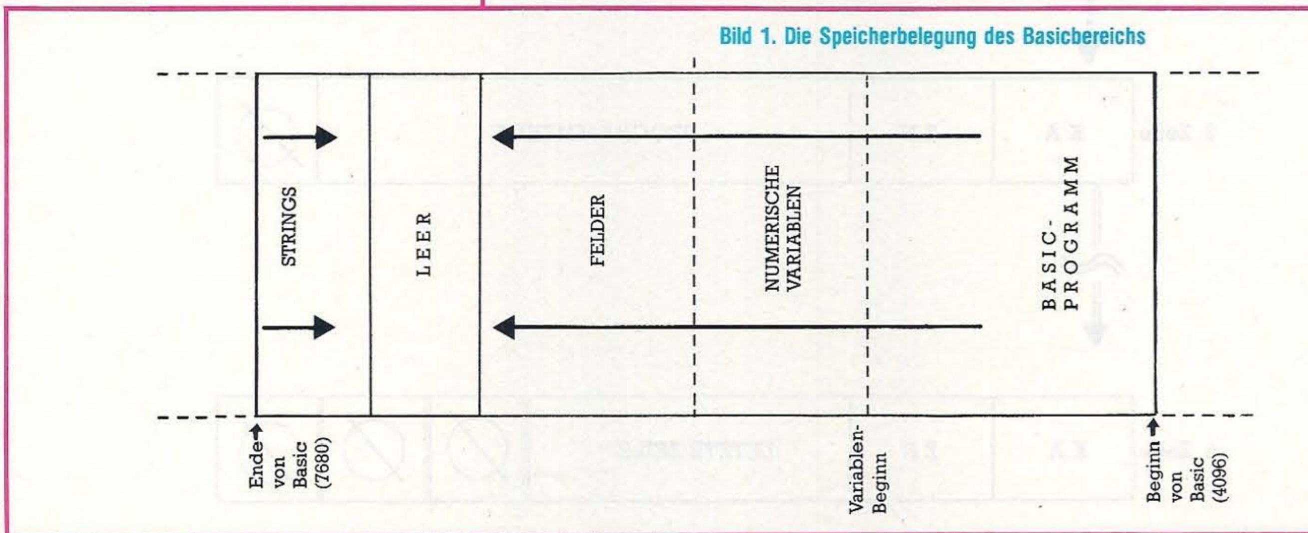


Bild 1. Die Speicherbelegung des Basicbereichs

Die andere Alternative der Platzbeschaffung ist etwas komplizierter. Sie wird nur bei erweitertem Speicher angewendet, um dort Sonderzeichen abzulegen. Um den Anfang des Programmspeichers von 4608 nach 7680 zu schieben, gibt man: POKE 44,30:POKE 30 * 256,0:NEW ein, denn 30 * 256 ist gerade 7680. Der zweite POKE-Befehl ist nötig, da am Anfang des Basicbereichs immer ein Nullbyte stehen muß.

Erste Hilfe — Basicprogramme retten nach NEW oder Reset

Schon oft wurden Verfahren beschrieben, um nach einem versehentlichen NEW das Basicprogramm wieder zurückzuholen. Doch wie funktionieren diese Verfahren? Um das zu verstehen, betrachten wir zunächst kurz den Aufbau eines Basicprogramms (Bild 2).

Am Kopf des Programms steht immer eine Null. Dann folgt die Adresse der nächsten Programmzeile (Koppeladresse) und die Zeilennummer. Danach kommt die eigentliche Programmzeile, die sich aus den sogenannten Tokens — den Basiccodenummern (Tabelle 2) — zusammensetzt. Am Ende dieser Zeile steht dann nochmals eine Null. Die

Tabelle 2. VC 20 Basic-Token. Die Codenummern 32 bis 95 entsprechen den ASCII-Codes. Nummern größer als 127 sind Token, also Abkürzungen für Basic-Befehle, die der Basic-Interpreter verwendet, um Speicherplatz zu sparen und die Verarbeitungsgeschwindigkeit zu erhöhen.

Code (Dezimal)	Zeichen/ Befehl	Code (Dezimal)	Zeichen/ Befehl	Code (Dezimal)	Zeichen/ Befehl	Code (Dezimal)	Zeichen/ Befehl
0	Zeilenende	66	B	133	INPUT	169	STEP
1-31	Leer	67	C	134	DIM	170	+
32	Space	68	D	135	READ	171	—
33	!	69	E	136	LET	172	*
34	"	70	F	137	GOTO	173	/
35	#	71	G	138	RUN	174	↑
36	$	72	H	139	IF	175	AND
37	%	73	I	140	RESTORE	176	OR
38	&	74	J	141	GOSUB	177	>
39	'	75	K	142	RETURN	178	=
40	(	76	L	143	REM	179	<
41	)	77	M	144	STOP	180	SGN
42	*	78	N	145	ON	181	INT
43	+	79	O	146	WAIT	182	ABS
44	'	80	P	147	LOAD	183	USR
45	—	81	Q	148	SAVE	184	FRE
46	.	82	R	149	VERIFY	185	POS
47	/	83	S	150	DEF	186	SQR
48	0	84	T	151	POKE	187	RND
49	1	85	U	152	PRINT#	188	LOG
50	2	86	V	153	PRINT	189	EXP
51	3	87	W	154	CONT	190	COS
52	4	88	X	155	LIST	191	SIN
53	5	89	Y	156	CLR	192	TAN
54	6	90	Z	157	CMD	193	ATN
55	7	91	[	158	SYS	194	PEEK
56	8	92	£	159	OPEN	195	LEN
57	9	93	]	160	CLOSE	196	STR$
58	:	94	↑	161	GET	197	VAL
59	;	95	"	162	NEW	198	ASC
60	<	96-127	Leer	163	TAB(	199	CHR$
61	=	128	END	164	TO	200	LEFT$
62	>	129	FOR	165	FN	201	RIGHT$
63	?	130	NEXT	166	SPC(	202	MID$
64	@	131	DATA	167	THEN	203-254	Leer
65	A	132	INPUT	168	NOT	255	

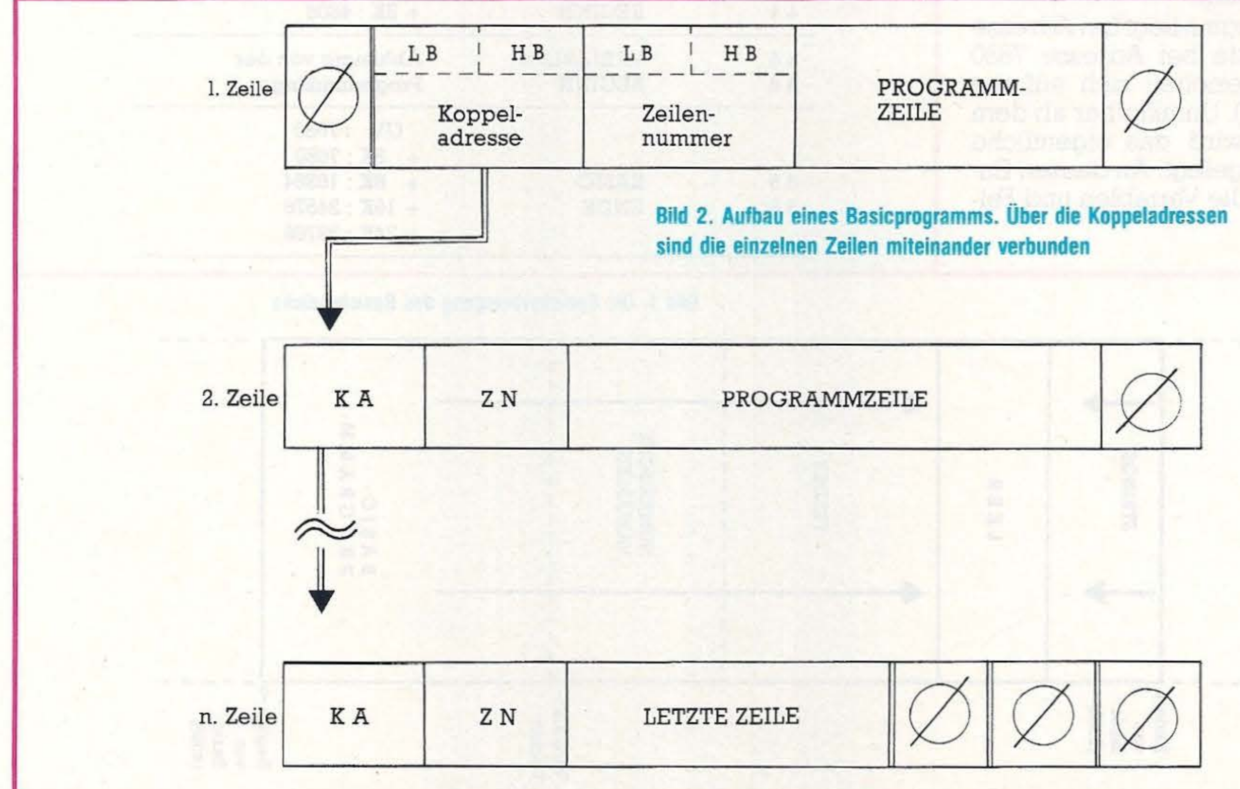


Bild 2. Aufbau eines Basicprogramms. Über die Koppeladressen sind die einzelnen Zeilen miteinander verbunden



nächste Zeile beginnt wieder mit einem Verbindungszeiger und der Zeilennummer. Das Programm wird mit drei Nullen abgeschlossen. Hieran schließen sich die Variablen an (vergleiche Bild 1).

Durch NEW oder durch einen RESET wird nicht das gesamte Programm, sondern nur der Variablenpointer (45/46) und die erste Koppeladresse gelöscht. Durch Rekonstruktion dieser beiden Zeiger kann das scheinbar verlorene Basicprogramm wieder benutzt werden.

Hier nun das »Rezept« zur Rekonstruktion:

- POKE (Basicanfang) + 2,1
  Basicanfang in GV = 4097
  + 3K = 1025
  + 8K = 4609
- SYS 50483:POKE 46,PEEK(35): POKE 45, PEEK (781)+2:CLR

Unbedingt wichtig ist hier die Reihenfolge der Befehle! Ferner darf während der gesamten Prozedur keine Variable definiert werden, da diese das gelöschte Programm überschreiben würde.

Die Funktionsweise ist relativ einfach. Die Unterprogrammroutine (SYS 50483) bindet die Programmzeilen neu und stellt dabei den ersten Verbindungszeiger wieder her. Sie übergibt dann in den beiden Speicherstellen (35 und 781) die Adresse des Variablenbeginns —2.

Die CHRGET-Routine

Die Zeropage ist in Maschinensprache besonders einfach zu adressieren. Aus diesem Grund sind hier oft benötigte Daten abgelegt. Doch die Seite Null beheimatet auch ein Unterprogramm aus dem Basicinterpreter namens CHRGET (CHaRacter GET, Tabelle 3). Diese Routine hat die Aufgabe, aus dem Basictext einzelne Zeichen oder Befehle zur Auswertung bereitzustellen. Sie befindet sich gerade deshalb im RAM-Speicher, weil sie einen veränderbaren 2-Byte-Zeiger enthält. Da die Routine bei jeder Ausführung eines Basicbefehls benutzt wird, bietet sich hier eine gute Möglichkeit, in den Ablauf einzugreifen, um damit den Befehlsvorrat zu erweitern. CHRGET endet mit einem Sprung zurück zur Befehlsauswertung. Da die CHRGET-Routine im RAM liegt, kann an dieser Stelle die Routine in das Befehlsauswertungsprogramm des Benutzers umgeleitet werden. Dort wird zuerst das CHRGET-Unterprogramm zu Ende geführt.

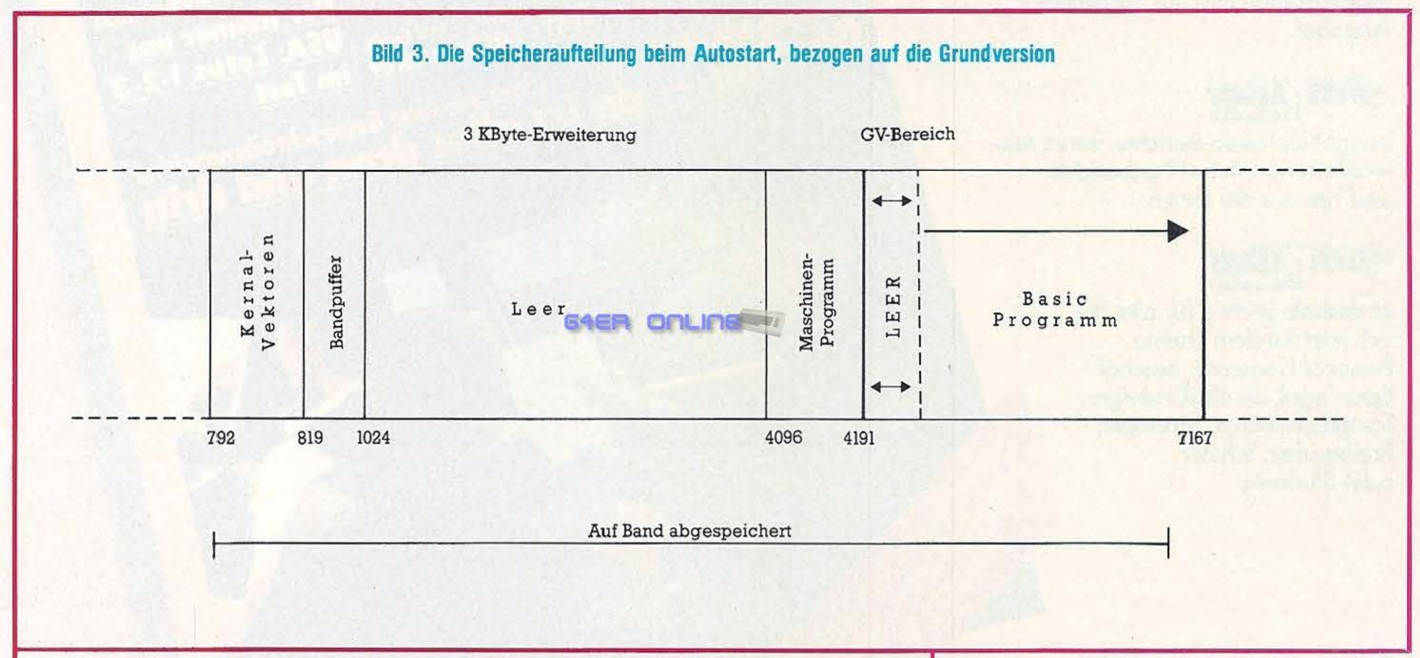


Bild 3. Die Speicheraufteilung beim Autostart, bezogen auf die Grundversion

Tabelle 3. Assemblerdarstellung der »CHRGET«-Routine

```
CHRGET:
 0073   INC $ 7A                ; erhöhe Lo-Byte
 0075   BNE $ 0079              ; Übertrag ?
 0077   INC $ 7B                ; ja, dann Hi-Byte erhöhen
 0079   LDA $ Pointer           ; hole Zeichen aus dem Basictext
 007C   JMP $ Befehlsauswertung
```

Tabelle 4. So könnte die »CHRGET«-Routine zu Ende geführt werden

```
Befehlsauswertung:
 1C00   CMP  #$ FF              ; Befehl PI ?
 1C02   BEQ  $ 1C16             ; Ja, dann RESET
 1C04   CMP  #$ 3A              ; Fortführung des alten CHRGET
 1C06   BCS  $ 1C15
 1C08   CMP  #$ 20
 1C0A   BNE  $ 1C0F
 1C0C   JMP  $ 0073
 1C0F   SEC
 1C10   SBC  #$ 30
 1C12   SEC
 1C13   SBC  #$ D0
 1C15   RTS
 1C16   JMP  $ FD22             ; eigene Befehlsverarbeitung:
                                RESET
```

Als Beispiel soll der bestehende Befehl π (Tokennummer 255) geändert werden. Das Befehlsauswertungsprogramm nach Tabelle 4 fragt ihn ab und verzweigt dann nach § 1C16, wo ein RESET ausgeführt wird (entspricht SYS 64802).

Man kann aber auch noch zusätzliche Parameter abfragen. Die nun folgende Änderung des π-Befehls steuert den Tongenerator, wobei drei Argumente und zwei Kommata geprüft werden müssen. Die Syntax des neuen Befehls ist aus Tabelle 5 ersichtlich.

Die Steuerungsroutine wird nun aus diesen vorgefertigten Modulen zusammengesetzt.

Zuerst der Baustein zum Abfragen von Argumenten (Tabelle 6).

Die ROM-Routine ($D79B) holt sich aus dem Basictext den numerischen Ausdruck und stellt ihn im X-Register zur Verfügung. Ist der Wert größer als 255, wird eine Fehlermeldung ausgegeben. Die Syntax unseres Befehls erlaubt aber nur Argumente zwischen 0 und 3. Daher wird nochmals eine Bereichsprüfung vorgenommen.

Als nächstes wird das Komma abgefragt (Tabelle 7). CHRGOT holt das laufende Zeichen aus dem Basictext und die Routine vergleicht es mit dem ASCII-Code für das Komma.

Man unterscheidet im übrigen zwischen CHRGET und CHRGOT. CHRGET ($0073) stellt den Zwei-Byte-Zeiger um eins hoch und lädt dann das neue Zeichen in das Akku. CHRGOT ($0079) hingegen holt lediglich das Zeichen.

Um den Soundbefehl zu komplettieren müssen noch die restlichen drei Module eingebaut werden. Wir wollen an dieser Stelle jedoch darauf verzichten, das im einzelnen auszuführen. Es sollte jetzt jedoch klar geworden sein, wie man eine Befehlserweiterung realisieren kann.

Für Assembler-Laien jetzt noch ein komplettes Befehlsprogramm. Es erweitert die bestehenden Kommandos um:

πO ≙ Old (Rekonstruktion),
π S Tongenerator, Höhe, Lautstärke ≙ Soundbefehl (wie oben)
π S Tongenerator, 0 ≙ Tongenerator abschalten,
π P Horizontal, Vertikal, "." oder
π P Horizontal, Vertikal, String ≙ Druck an einer spezifizierten Bildschirmstelle.

Das Ladeprogramm (Listing 1) lädt die Maschinenroutine automatisch in den richtigen Speicherbereich (abhängig von der Speichergröße) und gibt dann die Startadresse an. Zur Referenz ist das vollständige Assemblerprogramm als Listing 2 abgedruckt.

Die neuen Befehle können sowohl im Direktmodus, als auch im Programm verwendet werden. Benutzt man sie im Programm, so ist zu beachten, daß sie nie direkt nach der Zeilennummer stehen dürfen. So muß zum Beispiel der Befehl π S 1,240,15 mit Doppelpunkt im Programm stehen:
10 : π S1,240,15 oder
10 PRINT A$: π S1,240,15

Tabelle 7. Dieses Modul testet, ob ein Komma folgt.

1C22	JSR	$ 0079	;CHRGOT (laufendes Zeichen)
1C25	CMP	# 2C	;Komma?
1C27	BEQ	Weiter	;ja, dann verzweigen
1C29	JMP	$ CF08	; "Syntax error" ausgeben
1C2B		— Weiter —	

Listschutz für Basicprogramme:

Es wurden bereits mehrfach Methoden veröffentlicht, mit denen man Programme vor unbefugtem Kopieren schützen kann. Hierbei gibt es mehrere Alternativen:

1. Man verändert die Koppeladressen so, daß das Programm nicht listfähig ist, es jedoch normal mit RUN bedient werden kann.

2. Man verändert den LIST-Vektor mit POKE 774,34:POKE 755,253. Bei einem Listversuch löst dieser Vektor einen RESET aus und das Programm ist weg.

Listing 1. Basic-Lader zur Befehlserweiterung

```
100 REM ********************************
**
110 REM ****                          **
**
120 REM **** VC-20 BEFEHLSAUSWERTUNG  **
**
130 REM ****                          **
**
140 REM **** 1984 BY CHRISTOPH  SAUER **
**
150 REM ****                          **
**
160 REM ****    HUBERTUSSTRASSE 14    **
**
170 REM ****                          **
**
180 REM ****    8000  MUENCHEN  19    **
**
190 REM ****                          **
**
200 REM ********************************
**
210 FORT=0TO215:READD:P=P+D:NEXT:IFP<>23
243THENPRINT"FEHLER !!":END
220 POKE55,0:POKE56,PEEK(56)-1:CLR:A=PEE
K(56)
230 RESTORE:FORT=A*256TOA*256+215
240 READD
250 IFD=-1THEND=A
260 POKET,D
270 NEXT
280 PRINT"FERTIG. START MIT":PRINT"S
YS"A*256+203
290 PRINT"AKTIVIERUNG MIT 'A'"
300 GETA$:IFA$=""THEN300
310 IFA$<>"A"THENEND
320 SYSA*256+203:πS1,240,15:FORT=1TO500:
NEXT:πS1,0
330 DATA201,255,240,018,201
340 DATA058,176,013,201,032
350 DATA208,003,076,115,000
360 DATA056,233,048,056,233
370 DATA208,096,032,115,000
380 DATA201,079,208,003,076
390 DATA049,-01,201,080,208
400 DATA003,076,075,-01,201
410 DATA083,208,003,076,134
420 DATA-01,076,008,207,160
430 DATA001,145,043,032,051
440 DATA197,165,034,024,105
450 DATA002,133,045,165,035
460 DATA105,000,133,046,032
470 DATA096,198,076,116,196
480 DATA032,155,215,224,023
490 DATA144,003,076,072,210
500 DATA134,250,032,121,000
510 DATA201,044,240,003,076
520 DATA008,207,032,155,215
530 DATA224,022,144,003,076
540 DATA072,210,032,121,000
550 DATA201,044,240,003,076
560 DATA008,207,024,138,168
570 DATA166,250,032,010,229
580 DATA032,115,000,032,160
590 DATA202,076,193,-01,032
600 DATA155,215,224,004,144
610 DATA003,076,072,210,134
620 DATA250,032,121,000,201
630 DATA044,240,003,076,008
640 DATA207,032,155,215,134
650 DATA251,240,023,032,121
660 DATA000,201,044,240,003
670 DATA076,008,207,032,155
680 DATA215,224,016,144,003
690 DATA076,072,210,142,014
700 DATA144,166,250,165,251
710 DATA157,010,144,165,157
720 DATA016,003,076,116,196
730 DATA076,121,000,169,076
740 DATA133,124,169,000,133
750 DATA125,169,-01,133,126
760 DATA096
READY.
```

Tabelle 5. Die Syntax des neuen π-Befehls

Tongenerator Bereich 0 — 3	,	Tonhöhe Bereich 0 — 255	,	Lautstärke Bereich 0 — 15

Tabelle 6. Das Modul zur Argumentabfrage

1C16	JSR	$ D79B	; Numerisches Argument holen
1C19	CPX	#$ 04	; Ausdruck größer als 4 ?
1C1B	BCC	$ 1C20	; nein, dann verzweigen
1C1D	JMP	$ D248	; sonst Fehlermeldung ausgeben
1C20	STX	$ FA	; Wert in Zeropage ablegen

Diese und andere Schutzmaßnahmen haben den Nachteil, daß sie von »Hackern« innerhalb kurzer Zeit umgangen werden können. Es gibt zwar keinen hundertprozentigen Programmschutz, jedoch bietet die nachfolgend beschriebene Methode eine große Sicherheit. Bei diesem Verfahren läßt die Änderung eines Kernalvektors (Tastatureingabe 804/805) das Basicprogramm mit Hilfe einer Maschinenroutine nach Abschluß des Ladevorgangs automatisch starten.

Zunächst zur Verfahrensweise beim Autostart:

Schritt 1:
Eingabe des Ladeprogramms (Listing 3)

Schritt 2:
Programm und Prüfsumme testen (Achtung es zerstört sich selbst mit NEW) und abspeichern

Schritt 3:
POKE 44,A : POKE A*256,0 : NEW (A = 17 für die Grundversion; A = 19 bei Erweiterung ab 8 KByte; A = 5 Erweiterung + 3 KByte

Schritt 4:
Ladeprogramm laden und starten

Schritt 5:
Eigenes Programm nachladen

Schritt 6:
POKE 43,24 : POKE 44,3

Schritt 7:
POKE 792,91 : POKE 793,255 : POKE 808,114

Schritt 8:
POKE 804,0 : POKE 805,X : SAVE "...",1,1

X wird vom Ladeprogramm angegeben (X = 16 in Grundversion; X = 18 bei Erweiterung ab 8 KByte; X = 4 bei Erweiterung von 3 KByte). Die Befehle von Schritt 8 müssen unbedingt in einer Zeile stehen, sonst stürzt der Computer ab.

Das Ladeprogramm (oder einfacher der Lader) übernimmt die Abspeicherung des Maschinenprogramms, wobei er sich nach einer eventuell vorhandenen Speichererweiterung richtet.

Nun zur Bedienung:

Nach der Prüfsummenkontrolle ist die Speichergröße per Tastendruck einzugeben. Dadurch wird überprüft, ob man vor dem Laden POKE 44,A eingegeben hat, denn sonst würde sich das Programm selbst überschreiben. Dann wird nach der Anfangsadresse für das zu schützende Basicprogramm gefragt. Der Lader gibt der Einfachheit halber bereits die entsprechende Adresse vor. Man kann sie aber auch ändern, wodurch das Auffinden des Basic-

```
********************** CHRGET FORTSETZUNG
1D00  CMP #$FF       ; PI ?
1D02  BEQ $1D16      ; JA, DANN VERZWEIGEN
1D04  CMP #$3A       ; SONST CHRGET ZU ENDE FUEHREN
1D06  BCS $1D15
1D08  CMP #$20
1D0A  BNE $1D0F
1D0C  JMP $0073
1D0F  SEC
1D10  SBC #$30
1D12  SEC
1D13  SBC #$D0
1D15  RTS
1D16  JSR $0073      ; NAECHSTES ZEICHEN HOLEN
1D19  CMP #$4F       ; 'O' ?
1D1B  BNE $1D20      ; NEIN, DANN WEITER
1D1D  JMP $1D31      ; SONST VERZWEIGEN
1D20 CMP #$50        ; 'P' ?
1D22  BNE $1D27      ; NEIN, DANN WEITER
1D24  JMP $1D4B      ; SONST VERZWEIGEN
1D27  CMP #$53       ; 'S' ?
1D29  BNE $1D2E      ; NEIN, DANN FEHLER
1D2B  JMP $1D86      ; SONST VERZWEIGEN
1D2E  JMP $CF08      ; 'SYNTAX ERROR' AUSGEBEN
********************** O FUER REKONSTRUKTION
1D31  LDY #$01
1D33  STA ($2B),Y    ; AM BASICANFANG ABSPEICHERN
1D35  JSR $C533      ; BASICZEILEN NEU BINDEN
1D38  LDA $22
1D3A  CLC
1D3B  ADC #$02
1D3D  STA $2D
1D3F  LDA $23
1D41  ADC #$00
1D43  STA $2E        ; PROGRAMMENDE IN DEZ 43,44 ABSPEICHERN
1D45  JSR $C660      ; CLR DURCHFUEHREN
1D48  JMP $C474      ; ZURUECK IN DEN DIREKTMODUS
********************** P FUER POSITIONSDRUCK
1D4B  JSR $D79B      ; PARAMETER HOLEN
1D4E  CPX #$17       ; >23 ?
1D50  BCC $1D55      ; NEIN, DANN WEITER
1D52  JMP $D248      ; SONST FEHLERMELDUNG
1D55  STA $FA        ; HORIZONTAL POSITION ABSPEICHERN
1D57  JSR $0079      ; LFD. ZEICHEN HOLEN
1D5A  CMP #$2C       ; KOMMA ?
1D5C  BEQ $1D61      ; JA, DANN WEITER
1D5E  JMP $CF08      ; SONST SYNTAX ERROR
1D61  JSR $D79B      ; NAECHSTEN PARAMETER HOLEN
1D64  CPX #$16       ; >22 ?
1D66  BCC $1D6B      ; NEIN, DANN WEITER
1D68  JMP $D248      ; SONST FEHLERMELDUNG
1D6B  JSR $0079      ; NAECHSTES ZEICHEN HOLEN
1D6E  CMP #$2C       ; KOMMA ?
1D70  BEQ $1D75      ; JA, DANN WEITER
1D72  JMP $CF08      ; SONST SYNTAX ERROR
1D75  CLC            ; VORBEREITUNG FUER DAS UNTERPROGRAMM
1D76  TXA              CURSOR SETZEN
1D77  TAY
1D78  LDX $FA
1D7A  JSR $E50A      ; CURSOR AN POSITION (X/Y REG)
1D7D  JSR $0073      ; NAECHSTES ZEICHEN HOLEN
1D80  JSR $CAA0      ; STRING AUSWERTEN UND AUSGEBEN
1D83  JMP $20C1      ; ROUTINE ABSCHLIESSEN
********************** S FUER SOUND
1D86  JSR $D79B      ; PARAMETER HOLEN
1D89  CPX #$04       ; >4 ?
1D8B  BCC $1D90      ; NEIN, DANN WEITER
1D8D  JMP $D248      ; SONST FEHLERMELDUNG
1D90  STX $FA
1D92  JSR $0079      ; LFD. ZEICHEN HOLEN
1D95  CMP $#2C       ; KOMMA ?
1D97  BEQ $1D9C      ; JA, DANN VERZWEIGEN
1D99  JMP $CF08      ; SONST SYNTAX ERROR
1D9C  JSR $D79B      ; NAECHSTEN PARAMETER
1D9F  STX $FB
```

Listing 2. Das vollständige Assemblerprogramm zur Befehlserweiterung

```
1DA1  BEQ $1DBA    ; BEI 0 TONGENERATOR ABSCHALTEN
1DA3  JSR $0079    ; NAECHSTES ZEICHEN HOLEN
1DA6  CMP #$2C     ; KOMMA ?
1DA8  BEQ $1DAD    ; JA, DANN WEITER
1DAA  JMP $CF08    ; SONSR SYNTAX ERROR
1DAD  JSR $D79B    ; LETZTEN PARAMETER HOLEN
1DB0  CPX #$10     ; >16 ?
1DB2  BCC $1DB7    ; NEIN, DANN WEITER
1DB4  JMP $D248    ; SONST FEHLERMELDUNG
1DB7  STX $900E    ; LAUTSTAERKE
1DBA  LDX $FA
1DBC  LDA $FB
1DBE  STA $900A,X  ; TONHOEHE IN DEN TONGENERATOR
********************** ROUTINE ABSCHLIESSEN
1DC1  LDA $9D      ; FLAG DIREKTMODUS/PROGRAMM
1DC3  BPL $1DC8    ; FALLS PROGRAMM DANN VERZWEIGEN
1DC5  JMP $C474    ; DIREKTMODUS: READY EINSPRUNG
1DC8  JMP $0079    ; ZUR NORMALEN BEFEHLSBEARBEITUNG
********************** INITIALISIERUNG
1DCB  LDA #$4C
1DCD  STA $7C
1DCF  LDA #$00
1DD1  STA $7D
1DD3  LDA #$20
1DD5  STA $7E
1DD7  RTS
```

Listing 2. Assemblerprogramm zur Befehlserweiterung (Schluß)

```
100 REM ************************
110 REM ****                ****
120 REM **** BASIC-AUTOSTART ****
130 REM ****                ****
140 REM **** 1984 BY C.SAUER ****
150 REM ****                ****
160 REM **** HUBERTUSSTR. 14 ****
170 REM ****                ****
180 REM **** 8000 MUNCHEN 14 ****
190 REM ****                ****
200 REM ************************
210 FORT=1TO95:READD:P=P+D:NEXT
220 IFP<>9552THENPRINT"FEHLER !!":END
230 PRINT"AUTOSTART"
240 PRINT"---------"
250 PRINT"BITTE VERSION ANGEBEN:"
260 PRINT" 1= GV"
270 PRINT" 2= +3K"
280 PRINT" 3= >8K ":F=0
290 GETA$:IFA$=""THEN290
300 FORT=1TO3
310 IFA$=CHR$(T+48)THENF=T
320 NEXT
330 IFF=0THEN290
340 FORT=1TO4-F:PRINT"";:NEXT:PRINT""F
350 ONFGOTO360,370,380
360 S=17:GOTO390
370 S=5:GOTO390
380 S=19:GOTO390
390 AD=PEEK(44)
400 IFAD<>STHENPRINT"VOR DEM LADEN"
:PRINT"POKE 44,"S"EINGEBEN !":END
410 V=(S-1)*256:RESTORE
420 PRINT"ANFANGSADRESSE"
430 PRINT"FUER IHR PROG.  "V+96"
";:INPUTX
440 IFX<V+96THENPRINT"NICHT MOEGLICH !!
":GOTO420
450 X1=INT(X/256):X2=X-X1*256
460 FORI=VTOV+94
470 READD:IFD=-1THEND=S-1
480 IFD=-2THEND=X1
490 IFD=-3THEND=X2
500 POKEI,D:NEXT
510 PRINT"FERTIG. DER POKE FUER"
520 PRINT"805 IST"S-1". LADEN"
530 PRINT"SIE JETZT DAS HAUPT-"
540 PRINT"PROGRAMM NACH."
550 POKEX1*256+X2-1,0:POKE43,X2:POKE44,X
1:NEW
560 DATA169,014,141,036,003
570 DATA169,242,141,037,003
580 DATA169,194,141,020,003
590 DATA169,219,141,024,003
600 DATA169,010,141,137,002
610 DATA169,112,141,040,003
620 DATA169,-03,133,043,169
630 DATA-02,133,044,169,000
640 DATA198,043,168,145,043
650 DATA230,043,160,001,152
660 DATA145,043,032,051,197
670 DATA165,034,024,105,002
680 DATA133,045,165,035,105
690 DATA000,133,046,032,096
700 DATA198,162,005,189,083
710 DATA-01,157,118,002,202
720 DATA208,247,240,004,082
730 DATA085,078,013,169,004
740 DATA133,198,076,116,196
READY
```

Listing 3. Basicloader zum Autostart

programms nach einem RESET erschwert wird.

Zur Erklärung betrachten wir Bild 3. Es zeigt die Speicheraufteilung beim Autostart, bezogen auf die Grundversion. Das eigentliche Maschinenprogramm benötigt 95 Byte. Es liegt direkt am Basicbeginn (Adresse 4096). Dann folgt eine Lücke. Sie ist, wie bereits gesagt, nicht unbedingt nötig, aber sie erschwert etwaigen Raubkopierern das Auffinden des Programms. Hieran schließt sich das eigentliche Basicprogramm an.

So funktioniert der Autostart

Durch POKE 43,24 : POKE 44,3 wird der gesamte Bereich zwischen Adresse 792 und Programmende aufgezeichnet, wodurch sich die Ladezeit etwas erhöht.

Wie wir bereits gesehen haben, ist das Betriebssystem des VC 20 dank seiner Vektoren äußerst flexibel. Für den Programmschutz machen wir uns dabei folgende Zeiger zu Nutze:

1. NMI-Vektor, Adresse 792,793: Dieser Vektor stellt die Verbindung zwischen RESTORE-Taste und NMI-Routine her. Durch die Änderung (siehe Schritt 7) wird die RESTORE-Routine einfach übersprungen; die Taste ist quasi abgeschaltet.

2. STOP-Vektor, Adresse 808,809: Auch hier wird die bestehende Routine übersprungen.

Da dieser Vektorenbereich mit abgespeichert wird, ist, nachdem der Computer "FOUND" anzeigt, ein Stoppen des Computers nicht mehr möglich.

3. INPUT-Vektor, Adresse 804,805: Dieser Zeiger ist der eigentliche Dreh- und Angelpunkt des Autostarts. Er ist für die Tastatureingabe verantwortlich. Da er ständig durchlaufen wird, bewirkt POKE 804,0 : POKE 805,16 (bei geladener Autostartroutine) in der Grundversion einen Start des Basicprogramms. Ändert man den Zeiger hingegen vor dem Abspeichervorgang (wie in Schritt 8) geschieht vorläufig nichts, denn dann wird die Tastatur ja nicht benutzt.

Somit eignet sich dieser Vektor besonders gut für unseren Zweck. Denn solange sich der Computer mit dem Laden beschäftigt, ist die Tastatur »ruhig gestellt«. Der Zeiger wird so lange nicht benötigt, bis das Programm komplett geladen ist. Ist dies geschehen, springt das Betriebssystem über den INPUT-Vektor in die Autostartroutine, die ihrerseits (nach dem Rückstellen des Zeigers auf seinen ursprünglichen Wert) über RUN das Basicprogramm startet.

Damit auch alles wieder in den richtigen Speicherbereich geladen wird, dafür sorgt die Sekundäradresse bei SAVE " ",1,1.

Das Programm kann anschließend ganz normal mit LOAD geladen werden. Zum Schluß noch zwei Tips:

1. Wer besonders clever ist, der vernichtet alle »Spuren«, indem er die Maschinenroutine nach ihrer Benutzung im Basicprogramm löscht:
5 FOR T = (Startadresse) TO (Startadresse) + 95 : POKE T, RND(0) + 255 : NEXT
(Startadresse = 4096 in der Grundversion; = 1024 bei Erweiterung von 3 KByte; = 4608 bei Erweiterung ab 8 KByte

2. Bei einer Erweiterung von 8 KByte liegt ja bekanntlich der Bildschirmspeicher im Bereich zwischen Adresse 4096 und 4607. Somit wird er ebenfalls mit abgespeichert.

Soweit die erste Folge unseres Kurses. Im zweiten Teil wollen wir uns etwas näher mit der Zeropage beschäftigen und unter anderem zeigen, wie man mehrere Basicprogramme gleichzeitig im Speicher halten kann.

(Christoph Sauer/ev)

Daten

Wenn sie ein Spielmodul in Ihren Computer eingeschoben haben war das Ihr erster Kontakt zur Welt der EPROMs: Erasable Programmable Read-Only Memory. Das bedeutet, daß Daten, die in einem EPROM gespeichert sind, nur noch gelesen und erst durch ein bestimmtes Verfahren wieder gelöscht werden können.

Die Vorteile dieser Art der Datenspeicherung sind enorm:
— Das Programm ist sofort nach dem Einschalten des Computers im Speicher vorhanden.
— Besondere Funktionen, wie automatisches Laden des Directory nach dem Einschalten sind möglich.
— Problemlose Handhabung, da zum Laden eines Programms keine Kenntnisse von Programmiersprachen notwendig sind.

Die Nachteile liegen zum einen in den reltaiv hohen Kosten für die EPROMs (16 bis 160 Mark) und die notwendige Steckkarte (zirka 50 Mark). Zum anderen darin, daß es zu Überschneidungen im Speicherbereich kommen kann, wenn Programme von Diskette nachgeladen werden.

Wie funktioniert aber die Datenspeicherung auf EPROMs? Damit Sie die Daten des internen Speichers Ihres Computers in den Speicher eines EPROMs übertragen können, brauchen Sie einen »EPROM-Brenner«. Damit wird die Verbindung zwischen Computer und EPROM hergestellt und für die notwendige Programmierspannung gesorgt.

Das einzige, was jetzt noch fehlt, ist ein Programm, das die Datenübertragung steuert. Das Prinzip der EPROM-Programmierung beruht darauf, daß Ladungen in die Speicherzellen des EPROMs transportiert werden. Dabei wird nur zwischen zwei Ladungszuständen unterschieden: geladen und ungeladen. Die einzelnen Ladungszustände werden vom Computer entweder als logische 1 (high) oder logische 0 (low) interpretiert. Jede dieser Speicherzellen enthält somit ein Bit. Auch in EPROMs werden Daten in binärer Weise gespeichert. Die Speicherzellen eines neuen EPROMs sind normalerweise ungeladen (logische 1). Wird nun eine Programmierspannung (zwischen 12,5 und 25 V) auf eine dieser Zellen gelegt, so ändert sich ihr Potential, sie wird »geladen« (logisch 0). Dabei werden die am Datenbus des EPROMs anliegenden Daten in die durch den Adreßbus angegebene Adresse des EPROMs übernommen. Es können keine Bits, die durch eine bereits vorgenommene Programmierung auf logisch 0 (low) gesetzt sind, beim Programmieren wieder in logisch 1 (high) umgewandelt werden. Damit die Ladung der Speicherzellen auch nach dem Wegnehmen der Programmierspannung erhalten bleibt, ist jede Zelle von einer semipermeablen Isolierschicht umgeben. Entsprechend dem Programm wird so, immer acht Speicherzellen (ein Byte) auf einmal, das gesamte Programm aus dem Computer in den EPROM übertragen.

Natürlich hängt die Länge des übertragbaren Programms von der Speicherkapazität der verwendeten EPROMs ab. Die meistverwendeten Typen können dabei zwischen 2 und 16 KByte speichern, es gibt aber auch schon EPROMs mit 32 KByte. Ein Programm mit 8 KByte Länge kann so entweder auf einem 8-KByte-EPROM oder auf zwei 4-KByte-EPROMs gespeichert werden.

Wie das »Erasable« im Namen der EPROMs schon andeutet, ist der Speicherinhalt nicht auf alle Zeiten festgeschrieben. Der Inhalt eines EPROMs wird durch Bestrahlung mit ultraviolettem Licht wieder gelöscht. Dazu ist im Gehäuse des EPROMs ein rundes Fenster ausgespart, durch das die UV-Strahlen auf den Chip einwirken können. Beim Löschen wird die Isolationsschicht der Speicherstellen in beide Richtungen durchlässig, was eine Entladung zur Folge hat.

Der EPROM ist danach wieder programmierbar. Ein EPROM kann so zwischen 25 und 30 mal gelöscht und neu programmiert werden. Am billigsten ist es, die EPROMs zum Löschen einfach in die Sonne zu legen, leider auch am langsamsten. Schnell geht es mit einem speziellen EPROM-Löschgerät, das ist der teuerste Weg. Im Normalfall reicht eine einfache Höhensonne. Die Löschzeit beträgt dann, je nach Entfernung der EPROMs zur UV-Quelle, zwischen 15 und 25 Minuten.

Nun aber zur Praxis. Mit dem abgedruckten Assemblerlisting ist es möglich, jedes beliebige Basicprogramm in EPROMs zu brennen. Die Programme werden damit automatisch gestartet und sind gegen Programmabruch geschützt. Das Programm »EPROM-Maker« steht im Speicherbereich zwischen $8000 und $8100 und muß vor jedem anderen Programm in die Speicherzellen 0 bis 100 des EPROMs gebrannt werden. In Adresse $8004 beginnt die Meldung »CMB80«, die das Betriebssystem während seiner Initialisierungsroutine abfragt. Dadurch wird bewirkt, daß in die, in den Speicherstellen $8000 und $8001 abgelegten Adresse, ($800A) verzweigt wird. Dort wird dann der eigentliche Basic-Lader nach $C000 geladen. Vier Speicherstellen sind vor dem jeweiligen Brennen des EPROMs zu verändern. Dazu gehen Sie wie folgt vor:

1. Laden sie das zu brennende Basicprogramm ganz normal in den Speicher.
2. Ermitteln Sie den Inhalt der Speicherstellen 2049, 2050, 45 und 46 mittels PEEK.
3. Rechnen Sie die Werte in Hexadezimalzahlen um und schreiben Sie diese auf.
4. Laden Sie nun einen Monitor
5. Laden Sie den EPROM-Maker und starten Sie den Monitor
6. Schreiben Sie den Wert der Speicherzelle 2049 in den LDA-Befehl in 80AC, den von 2050 in 80B1, den von 45 in 80B6 und den von 46 in 80BA.

Sie haben dann ein individuelles Startprogramm für Ihr Basicprogramm.

7. Laden Sie die Treibersoftware für Ihren EPROM-Brenner.
8. Übertragen Sie Ihren EPROM-Maker in die ersten 100 Speicherzellen des EPROMs.
9. Nun können Sie Ihr Basicprogramm (ab $0800) mit EPROM-Start 100 brennen. Reine Maschinenprogramme können Sie so natürlich auch brennen, wenn Sie diese zuvor in DATA-Zeilen mit Ladeschleife umgewandelt haben.

Das Programm ist auf die Platine von M. Frank zugeschnitten, so daß bis zu 16 KByte lange Programme problemlos gebrannt werden können. Wer über die Platine nicht verfügt, ist leider auf 8 KByte beschränkt. Wenn Sie die Autostartfunktion nicht benötigen, brauchen Sie den EPROM-Maker für Maschinenprogramme natürlich nicht. Hier genügt es, den EPROM einfach ab Speicherzelle 0 mit dem gewünschten Speicherbereich zu brennen und durch SYS (Startadresse) zu starten.

(A. Wängler/M. Frank/gk)

brennerei

Nicht aus Schottland und auch nicht trinkbar, aber dennoch gehaltvoll: EPROMs, fest programmierbare Speicher für Ihren C 64. Wie man sich eigene Programm-Module herstellt und was dahinter steckt, zeigt dieser Bericht.



Assemblerlisting zum Brennen eines EPROMs

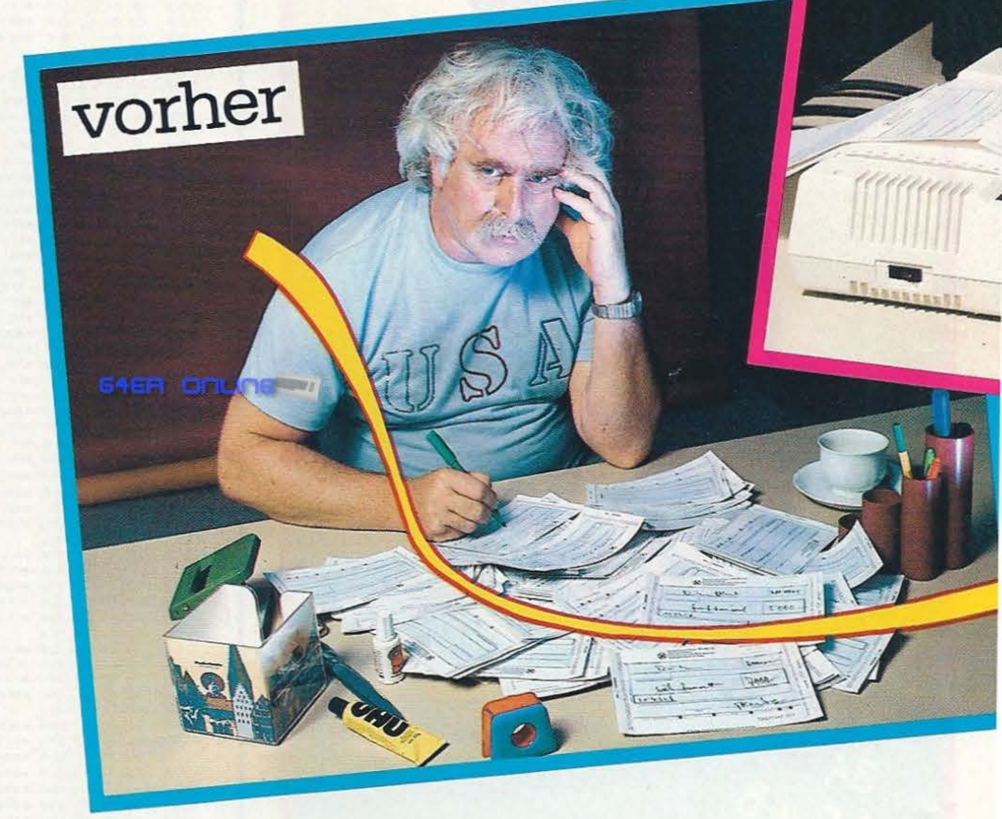
```
., 8000  0A        ASL            Autostartadresse des Programmes
., 8001  80        ???            = 800A
., 8002  94 E3     STY $E3,X      NMI Adresse (Restore Taste)
., 8004  C3        ???            Autostartkennung (CBM80)
., 8005  C2        ???            C3 C2 CD 38 30
., 8006  CD 38 30  CMP $3038
., 8009  00        BRK
., 800A  A0 C0     LDY #$C0       Eigentlicher Basicloader (steht ab 8029)
., 800C  84 FF     STY $FF        wird nach $C000 geladen
., 800E  A0 80     LDY #$80       ↑
., 8010  84 FD     STY $FD        ↑
., 8012  A0 29     LDY #$29       ↑
., 8014  84 FC     STY $FC        ↑
., 8016  A0 00     LDY #$00       ↑
., 8018  84 FE     STY $FE        ↑
., 801A  B1 FC     LDA ($FC),Y    ↑
., 801C  91 FE     STA ($FE),Y    ↑
., 801E  C0 D6     CPY #$D6       ↑
., 8020  F0 04     BEQ $8026      ↑
., 8022  C8        INY            ↑
., 8023  4C 1A 80  JMP $801A      ↑
., 8026  4C 00 C0  JMP $C000      Verzweigt zum Basicloader
., 8029  A2 05     LDX #$05       BASICLOADER TEIL 1
., 802B  8E 16 D0  STX $D016      Hier wird der 1.Teil des Basic
., 802E  A2 00     LDX #$00       Programmes in den Basicspeicher
., 8030  86 FC     STX $FC        (ab $0800) geladen
., 8032  86 FE     STX $FE        ↑
., 8034  A2 08     LDX #$08       ↑
., 8036  86 FD     STX $FD        ↑
., 8038  A2 81     LDX #$81       ↑
., 803A  86 FF     STX $FF        ↑
., 803C  A0 00     LDY #$00       ↑
., 803E  B1 FE     LDA ($FE),Y    ↑
., 8040  91 FC     STA ($FC),Y    ↑
., 8042  E6 FC     INC $FC        ↑
., 8044  A6 FC     LDX $FC        ↑
., 8046  E0 00     CPX #$00       ↑
., 8048  D0 02     BNE $804C      ↑
., 804A  E6 FD     INC $FD        ↑
., 804C  E6 FE     INC $FE        ↑
., 804E  A6 FE     LDX $FE        ↑
., 8050  E0 00     CPX #$00       ↑
., 8052  D0 EA     BNE $803E      ↑
., 8054  E6 FF     INC $FF        ↑
., 8056  A6 FF     LDX $FF        ↑
., 8058  E0 A0     CPX #$A0       ↑
., 805A  D0 E2     BNE $803E      ↑
., 805C  A9 01     LDA #$01       Platine wird auf zweiten 8kb Block
., 805E  8D 00 DE  STA $DE00      umgeschalten
., 8061  A2 80     LDX #$80       BASICLOADER TEIL 2
., 8063  86 FF     STX $FF        ↑
., 8065  84 FE     STY $FE        ↑
., 8067  B1 FE     LDA ($FE),Y    ↑
., 8069  91 FC     STA ($FC),Y    ↑
., 806B  E6 FC     INC $FC        ↑
., 806D  A6 FC     LDX $FC        ↑
., 806F  E0 00     CPX #$00       ↑
., 8071  D0 02     BNE $8075      ↑
., 8073  E6 FD     INC $FD        ↑
., 8075  E6 FE     INC $FE        ↑
., 8077  A6 FE     LDX $FE        ↑
., 8079  E0 00     CPX #$00       ↑
., 807B  D0 EA     BNE $8067      ↑
., 807D  E6 FF     INC $FF        ↑
., 807F  A6 FF     LDX $FF        ↑
., 8081  E0 A0     CPX #$A0       ↑
., 8083  D0 E2     BNE $8067      ↑
., 8085  A2 02     LDX #$02       1. 8KB Block der Platine wird wieder
., 8087  8E 00 DE  STX $DE00      eingeschalten
., 808A  78        SEI
., 808B  20 A3 FD  JSR $FDA3      Initialisierung des Betriebssystems
., 808E  20 50 FD  JSR $FD50      ↑
., 8091  20 15 FD  JSR $FD15      ↑
., 8094  20 5B FF  JSR $FF5B      ↑
., 8097  20 18 E5  JSR $E518      ↑
., 809A  58        CLI            ↑
., 809B  20 53 E4  JSR $E453      Initialisierung von Basic
., 809E  20 BE E3  JSR $E3BE      ↑
., 80A1  20 22 E4  JSR $E422      ↑
., 80A4  A2 FB     LDX #$FB       ↑
., 80A6  9A        TXS            ↑
., 80A7  A9 EF     LDA #$EF       ↑
., 80A9  8D 28 03  STA $0328      ↑
., 80AC  A9 0E     LDA #$0E       Hier werden die aktuellen Programmzeiger
., 80AE  8D 01 08  STA $0801      geschrieben ,die fuer jedes Programm neu
., 80B1  A9 08     LDA #$08       ermittelt werden muessen. (Siehe
., 80B3  8D 02 08  STA $0802      Text)PEEK 2049,PEEK 2050 => 2.
., 80B6  A9 D3     LDA #$D3       Programmzeile
., 80B8  85 2D     STA $2D
., 80BA  A9 46     LDA #$46       PEEK 45 ,PEEK 46 => Beginn der Variablen
., 80BC  85 2E     STA $2E
., 80BE  A9 8B     LDA #$8B
., 80C0  8D 00 03  STA $0300
., 80C3  A9 E3     LDA #$E3
., 80C5  8D 01 03  STA $0301
., 80C8  A9 E3     LDA #$E3
., 80CA  8D 07 03  STA $0307
., 80CD  8D 33 03  STA $0333
., 80D0  A9 94     LDA #$94
., 80D2  8D 06 03  STA $0306
., 80D5  8D 32 03  STA $0332
., 80D8  A9 02     LDA #$02
., 80DA  8D 00 DE  STA $DE00
., 80DD  A9 52     LDA #$52       Ablegen des Befehls RUN + chr$(13) im
., 80DF  8D 77 02  STA $0277      Tastaturpuffer und Einsprung in die Basic
., 80E2  A9 55     LDA #$55       Eingaberoutine (Autostart)
., 80E4  8D 78 02  STA $0278
., 80E7  A9 4E     LDA #$4E
., 80E9  8D 79 02  STA $0279
., 80EC  A9 0D     LDA #$0D
., 80EE  8D 7A 02  STA $027A
., 80F1  A9 04     LDA #$04
., 80F3  85 C6     STA $C6
., 80F5  4C 74 A4  JMP $A474
., 80F8  00        BRK
., 80F9  00        BRK
., 80FA  00        BRK
., 80FB  00        BRK
., 80FC  00        BRK
., 80FD  00        BRK
., 80FE  00        BRK
., 80FF  00        BRK
., 8100  01 00     ORA ($00,X)
```

Geregelter Zahlungsve…

Bargeldlos bezahlen ist eine feine Sache, doch das Ausfüllen der Formulare kann manchmal ganz schön lästig fallen. Sind nacheinander mehrere Überweisungsträger zu »beschreiben«, artet es schon in »Arbeit« aus.

Auch der Lehrer Manfred Bräuler mußte diese Erfahrung machen. Es war einfach lästig — immer wenn er Überweisungen tätigte, spannte er das Fomular in seine Schreibmaschine ein und mußte zum wiederholten Male eigene Kenndaten, wie Absender und Kontonummer eintragen. Vertippte er sich dabei, landete das für seine Auffassung mißverständlich ausgefüllte Formular im Papierkorb, denn auf der Durchschrift war eine Korrektur nicht eindeutig identifizierbar. Das waren manchmal Stunden, die eher einer langweiligen Beschäftigungsübung glichen als einer sinnvollen Tätigkeit.

Heute geht Manfred Bräuler ganz anders vor, denn mittlerweile hat er »den Stein des Weisen« gefunden. Auf seinem C 64 schrieb er ein Programm, das ihm weitgehend Doppeleingaben erspart. Hat sich ein Stapel von Rechnungen gesammelt, macht er sich ans Werk. Das Überweisungsformular wird in den Drucker — einen Epson RX-80 — eingespannt, das Programm geladen und los geht's. Die eigene Kontonummer und Absenderdaten sind im Programm festgelegt (Zeile 70 und 80) und müssen nicht mehr bei jeder Überweisung neu eingegeben werden. Die Daten von Empfängern, denen Manfred Bräuler häufiger Geld zahlen muß, hat er fest abgespeichert (Zeile 95) und erspart sich damit auch hier unnötige Tipparbeit. Über die Zeile 800 werden die Namen der Empfänger numeriert am Bildschirm ausgegeben. Wieviele Empfänger abgespeichert werden können, hängt allein von der Größe des Arbeitsspeichers ab. Aus der vorgeschlagenen Liste wählt Manfred Bräuler dann den gewünschten Empfänger aus, indem er dessen Nummer eingibt (Zeile 1000). Die Zeile 1200 führt dann weiter zu dem jeweils gewünschten Datensatz: Es erscheint die vollständige Anschrift und Bankverbindung des Empfängers. Bei einem nicht gespeicherten Empfänger erfolgt ab Zeile 100 die Abfrage der notwendigen Daten.



Ob vorher gespeichert oder nicht: Weiter geht's in Zeile 150; hier gibt Manfred Bräuler ein, warum er das Geld überweist, ob es beispielsweise die Miete ist oder die Bezahlung der letzten Autoreparatur. Ab Zeile 235 wird alles noch einmal am Bildschirm gezeigt, wobei die Möglichkeit eingeräumt ist, Eingabefehler zu korrigieren (ab Zeile 420). Sind alle Eingaben als richtig bestätigt worden, wird die Überweisung ausgedruckt. Und noch ein »Komfort« ist eingebaut: Über Zeile 630 kann bestimmt werden, ob eine weitere Überweisung auszudrucken ist, und wenn ja, ob sie an denselben Empfänger geht.

Manfred Bräuler setzt seinen C 64 nicht nur ein, um notwendige Arbeiten zu rationalisieren. Das Überweisungsprogramm schrieb er nach eigenen Aussagen in erster Linie aus Spaß an der Freud. Und Freude hatte er schon häufig mit dem Computer. Das war auch bezweckt, als er sich im August letzten Jahres zunächst die Grundeinheit kaufte. Er



wollte Basic lernen, um dann Programme zur Arbeitserleichterung und zum Spielen zu schreiben. Mittlerweile erweiterte er sein System, in zwei »Ausbaustufen«. Zunächst beendete er das Spiel mit der Datasette; sie genügte seinen Ansprüchen nicht mehr: Speicher- und Ladevorgänge dauerten zu lange und nach seiner Ansicht war sie auch zu unzuverlässig; einige Programme, die er sich gekauft hatte, liefen einfach nicht. So legte er sich eine Floppy zu. Die zweite Erweiterung kam im Dezember ins Haus: ein Drucker. Manfred Bräuler war es leid, alles nur auf dem Bildschirm anschauen zu können, er wollte seine Daten und Programme auch schwarz auf weiß sehen.

Momentan ist ein Textverarbeitungsprogramm in Arbeit, wobei Manfred Bräuler auch gleichzeitig seine Programmierkenntnisse erweitern will. Auch bei der Schularbeit läßt sich der Lehrer von seinem C64 entlasten: Die immer wieder anfallenden Zeugnisse werden per Computer ausgestellt.

Einen kleinen Haken hat die ganze Sache, denn der Computer zeigt sich in einer Hinsicht als regelrechter Dieb: Er stiehlt die Zeit, die vorher der Familie gehörte.

(Manfred Bräuler/kg)

Das Listing finden Sie auf Seite 166

X 01630841

Überweisungsauftrag an 660 908 00

Badische Beamtenbank eG
Postfach 1428 · 7500 Karlsruhe 1

D

Bitte kräftig und deutlich durchschreiben

Hier ● festhalten!
Bitte den ausgefüllten Formularsatz am linken Rand festhalten und ruckartig nach rechts auseinanderziehen.
420 033 dgverlag 20

Empfänger (Name und Anschrift)
FIRMA BAERWIND LAMPERTHEIM INDUSTRIESTR. 115

Konto-Nr. des Empfängers
5678912.

bei (Bank usw.) – oder ein anderes Konto des Empfängers[1]
KREIS- U. STADTSPARKASSE WORMS

Bankleitzahl
66090800

Verwendungszweck (nur für Empfänger)
RECHNUNG VOM 11.11.1911
UNGÜLTIGES M U S T E R

DM
****11,00

Betrag in Buchstaben (unter 1000 DM entbehrlich). Freies Feld durchstreichen.
ELFUNDZWÖLFZIG**********************************

Konto-Nr. des Auftraggebers
1 234 567

Auftraggeber
M. BRÄULER, 6840 LAMPERTHEIM, WORMSER STR. 118

[1] Soll die Überweisung auf ein anderes Konto ausgeschlossen sein, so sind die Worte „oder ein anderes Konto des Empfängers" zu streichen

10.02.1984

Datum

Unterschrift

Bitte die für den Auftraggeber bestimmte Einlage vor Weitergabe an die Bank entnehmen!

Mehrzweckfeld | Konto-Nr. | Betrag | Bankleitzahl | Text

583[redacted]H

66090800Ꭾ 20H

Bitte dieses Feld nicht beschriften und nicht bestempeln

Tippfehler beim ausgefüllten Formular — so gut wie unmöglich

```
10 REM*********************************
20 REM* DRUCK VON BANKUEBERWEISUNGEN *
25 REM*   CBM 64 - EPSON RX-80F/T    *
30 REM*   MANFRED BRAEULER  1984     *
35 REM*   WORMSER STR. 118           *
40 REM*   6840 LAMPERTHEIM           *
45 REM*********************************
49 :
50 PRINTCHR$(147)      : REM CLR/HOME
55 POKE 53280,6: REM BILDSCHIRMRAND BLAU
,KANN ENTFALLEN!
60 LE$="*********************************
***************":REM ENTWERTUNG
70 KA$="5 999 085          ":REM KONTON
R. DES ABSENDERS
80 A$="M. BRIULER, 6840 LAMPERTHEIM, WOR
MSER STR. 118"     : REM ABSENDER
90 PRINT"          BANKUEBERWEISUNG":PRIN
T"FOLGENDE EMPFAENGER SIND IM SPEICHER:"
95 GOTO 800: REM GESPEICHERTE EMPFAENGER
100 PRINT:PRINT"   UEBERWEISUNGSAUFTRAG"
110 PRINT:PRINT:PRINT"EMPFAENGER (MAX. 5
0 ZEICHEN)
115 INPUTE$: E=LEN(E$)
120 IF E>50 THEN PRINT"ZEILE ZU LANG, BI
TTE NEUE EINGABE!": GOTO 115
130 INPUT"KONTONUMMER ";K$:K=LEN(K$)
140 INPUT"BANKLEITZAHL";BL$
145 PRINT"BEI BANK (MAX. 45 ZEICHEN)
147 INPUTB$
148 IF LEN (B$)>45THEN PRINT"ZEILE ZU LA
NG,BITTE NEUE EINGABE!":GOTO 147
150 PRINT"VERWENDUNGSZWECK     (ZWEI ZEIL
EN MIT JE MAX. 45 ZEICHEN MOEGLICH)"
155 INPUTV1$
160 IF LEN (V1$)>45THEN PRINT"ZEILE ZU L
ANG,BITTE NEUE EINGABE!":GOTO 155
170 PRINT"ZWEITE ZEILE (LEER IST 'RETURN
')"
180 INPUTV2$:V=LEN(V2$)
190 IF V>45 THEN PRINT"ZEILE ZU LANG, BI
TTE NEUE EINGABE!": GOTO 180
200 PRINT"BETRAG IN BUCHSTABEN (MAX. 45
ZEICHEN)
210 INPUTBE$:B=LEN(BE$)
215 IF B>45 THEN PRINT"ZEILE ZU LANG, BI
TTE NEUE EINGABE!": GOTO 210
220 PRINT"BETRAG ALS ZAHL (DM 'RETURN' P
FENNIGE    >KEIN KOMMA<)": INPUTDM$,PF$
230 INPUT"DATUM";D$
234 :
235 REM*BILDSCHIRMAUSGABE ZUR KONTROLLE
240 PRINTCHR$(147)
250 PRINT:PRINT"        UEBERWEISUNG"
300 PRINT:PRINT"EMPFAENGER          :  ";E
$
320 PRINT:PRINT"KONTONUMMER         :  ";K
$
330 PRINT"BANKLEITZAHL       :  ";BL$
340 PRINT"BEI                :  ";B$
350 PRINT:PRINT"VERWENDUNGSZWECK  :  ";V
1$;" ";V2$
370 PRINT:PRINT"BETRAG IN WORTEN  :  ";B
E$
390 PRINT:PRINT"BETRAG ALS ZAHL   :  ";D
M$;",";PF$
400 PRINT:PRINT"KNTONR. AUFTRAGG. :  ";K
A$
410 PRINT:PRINT"AUFTRAGGEBER      :  ";A
$
415 PRINT:PRINT"DATUM             :  ";D
$
420 PRINT:PRINT"  ";CHR$(18);"  ALLES RI
CHTIG <J/N>";CHR$(146): REM RVS ON/OFF
430 GET X$: IF X$<>"J"ANDX$<>"N" THEN 43
0
440 IF X$="N" THEN 50
448 :
449 REM*DRUCKERAUSGABE
450 IF X$<>"J" THEN 430
500 PRINTCHR$(147)
510 PRINT:PRINTCHR$(18);"  FORMULAR IN D
RUCKER EINLEGEN!";CHR$(146)
520 PRINT:PRINT"DRUCKERSTART DURCH DRUEC
KEN EINER       BELIEBIGEN TASTE!"
530 GET T$: IFT$="" THEN 530
540 OPEN1,4:PRINT#1,CHR$(27);"M";: REM**
*   SCHOENSCHRIFT AN
550 PRINT#1,E$;TAB(55-E);BL$
555 PRINT#1,
560 PRINT#1,:PRINT#1,K$;TAB(20-K);B$
570 PRINT#1,:PRINT#1,V1$
580 PRINT#1,V2$;TAB(52-V);"****";DM$;","
;PF$
585 PRINT#1,
590 PRINT#1,BE$;LEFT$(LE$,(47-B))
595 PRINT#1,
600 PRINT#1,KA$;A$
610 PRINT#1,:PRINT#1,TAB(20);D$
615 PRINT#1,:PRINT#1,:PRINT#1,
620 PRINT#1,CHR$(27);CHR$(64): REM ***
DRUCKER NORMIEREN
625 CLOSE1
628 :
629 REM***WIEDERHOLUNG?
630 PRINT:PRINT"NOCH EINE UEBERWEISUNG <
J/N>?"
640 GET  UE$: IF UE$<>"J"ANDUE$<>"N" THE
N 640
645 IF UE$="N" THEN 670
650 IF UE$="J" THEN PRINT"GLEICHER EMPFA
ENGER    <J/N>?"
655 GET EM$: IF EM$<>"J"ANDEM$<>"N" THEN
655
660 IF EM$="J"THEN PRINTCHR$(147):GOTO15
0
665 IF EM$="N"THEN RUN
670 PRINTCHR$(147)
680 PRINT:PRINT"    PROGRAMMENDE":END
689 :
790 REM***AUFLISTUNG DER GESPEICHERTEN E
MPFAENGER MIT VERZWEIGUNG AB 1000
795 :
800 PRINT:PRINT"2< DR. GRAUNWALD"
810 PRINT"3< FIRMA BAERWIND
990 PRINT"1< EMPFAENGER IST NICHT GESPEI
CHERT
994 :
995 REM*WEITERE EMPFAENGER KOENNEN NACH
DEM GLEICHEN MUSTER GESPEICHERT WERDEN!
996 REM*AB1200 FOLGEN DIE GESPEICHERTEN
EMPFAENGERDATEN                        *
997 :
1000 PRINT:PRINTCHR$(18);"BITTE NUMMER D
ES EMPFAENGERS EINGEBEN";CHR$(146)
1020 GET NR$: IF NR$=""THEN1020
1030 N=VAL(NR$)
1040 PRINTCHR$(147)
1200 ON N  GOTO    100,1210,1220
1210 E$="DR. MED. GRAUNWALD LAMPERTHEIM
BARTSTR. 112": E=43
1212 K$="12345":K=5:BL$="22258909":B$="D
RESDNER BANK LAMPERTHEIM FILIALE MITTE"
1215 GOTO 150
1220 E$="FIRMA BAERWALD LAMPERTHEIM INDU
STRIESTR. 115":E=44
1222 K$="5678912":K=7:BL$="66090800":B$=
"KREIS- U. STADTSPARKASSE WORMS"
1225 GOTO 150
READY.
```

Listing: Druck von Banküberweisungen für C 64



Mitmachen, aber wie?

Wie schicke ich meine Programme ein?

Die 64'er-Redaktion freut sich über jeden Beitrag unserer Leser. Die Erfahrungen bei unseren Schwesterzeitschriften haben aber gezeigt, daß viele Einsender nicht genau wissen, in welcher Form sie ihre Manuskripte einsenden sollen. Die unten aufgeführten Punkte stellen keine »Richtlinien« dar. Dennoch sollte sich jeder, der ein Programm oder einen Artikel einsenden will, an ein gewisses Schema halten. Dies erleichtert zum einen die Arbeit der Redaktion, zum anderen kommt es auch Ihnen selbst zugute, da wir vollständige Listings oder Artikel schneller veröffentlichen können. Folgende Kriterien sind also generell zu beachten.

1. Auf der ersten Seite des Anschreibens sollten der Name, die vollständige Anschrift mit Telefonnummer sowie das Einsendedatum stehen.

2. In der »Betreffzeile« tragen Sie die genaue Spezifikation des verwendeten Computers und falls erforderlich, die Basic-, ROM- oder DOS-Versionen sowie die Speicherkonfigurationen ein. Der Titel des Artikels sollte ebenfalls daraus ersichtlich sein (auch für eventuelle Nachträge).

3. Im darauffolgenden Text können Sie Wesentliches zu Ihrer Person, zur Entstehungsgeschichte des Programms/Artikels, der Absicht, der Vorteile gegenüber anderen Programmen oder Methoden, der Eigenschaften und so weiter erläutern.

4. Auf der nächsten Seite beginnt die eigentliche Programmbeschreibung. Diese sollte nach Möglichkeit mit der Schreibmaschine geschrieben werden oder als Computerausdruck vorliegen. Den Text bitte mit mindestens eineinhalb oder doppeltem Zeilenabstand verfassen. Am linken und rechten Rand mindestens drei Zentimeter Freiraum für Korrekturen und Bemerkungen lassen.

5. Diese und alle nachfolgenden Seiten sollten durchnumeriert sein und in der Kopfzeile jeweils den Titel des Programms und den Namen des Autors enthalten.

6. Der Überschrift des Artikels schließen sich zwei oder drei einleitende Sätze an, welche die wesentlichen Punkte des Textes zusammenfassen.

Der Text selbst sollte in etwa folgenden Aufbau aufweisen:

— Angaben auf welchem Computer das Programm lauffähig ist sowie welche Erweiterungen und Peripherie notwendig sind

— ausführliche Beschreibung der Programmfunktion (mit Verweisen auf Ein-/Ausgabebeispielen wie Grafiken, Bildschirmfotografien, Hardcopys oder Diagrammen)

— detaillierte Programmbeschreibung (mit Verweisen auf Programmablaufplan, Variablendefinition, Startadressen der einzelnen Unterprogramme, Beschreibung wichtiger Programmzeilen etc.)

— eventuelle Umsetzung auf andere Basic-Dialekte oder Computer

7. Die genauen Lade- und Abspeicherschritte des Programms und der im Programm vorkommenden Routinen sollten dokumentiert sein.

8. Listings aus reprotechnischen Gründen nur als Original (keine Kopien) auf weißem, unliniertem Papier mit neuwertigem Farbband gedruckt einsenden. In den Listings dürfen grundsätzlich keine handschriftlichen Eintragungen stehen.

9. In den Kopfzeilen des Programms bitte den Titel desselben, die Computerkonfiguration, den eigenen Namen und die Adresse mit Telefonnummer eintragen (es soll vorkommen, daß sich Listings und Manuskripte verselbständigen, und mit beiden allein läßt sich wenig anfangen).

REM-Zeilen im Programm dienen der Übersichtlichkeit und sollten, falls nicht speicherkritische Aspekte dagegensprechen, immer zur Strukturierung eingesetzt werden (siehe u. a. »Sauberes Programmieren«).

10. Um das Eintippen für andere zu erleichtern, sollten Sie CHR$(X)-Werte und TAB(X) oder SPC(X) anstatt Cursor-Manipulationen für die Ausgabeformatierung verwenden. So ist die Befehlssequenz FOR I=1 TO 6:PRINT:NEXT zur Erzeugung von sechs Carriage Returns leichter einzutippen und auf andere Basic-Computer wesentlich einfacher zu übertragen. Und ist es nicht auch übersichtlicher statt einem Dutzend Cursor-Rechts-Symbolen einfach SPC(12) zu benutzen? Überprüfen Sie Ihr Programm einmal hinsichtlich dieser »Kleinigkeiten«.

11. Da wir (in Ihrem eigenen Interesse) nur getestete Programme veröffentlichen wollen, legen Sie bitte unbedingt eine Diskette oder Kassette, auf der das betreffende Programm mit mindestens einer Sicherheitskopie abgespeichert ist, bei. Auf der Diskette/Kassette und deren Umhüllung unbedingt den Namen mit vollständiger Adresse und Computerbezeichnung vermerken.

12. Wollen Sie mehrere Programme/Artikel gleichzeitig einsenden, so trennen Sie die Programme/Artikel nach dem oben aufgezeigten Schema. Die Einsendung mehrerer Disketten/Kassetten ist hingegen nicht notwendig.

13. Artikel können beliebig lang sein — von einzeiligen Routinen bis zu Serien über mehrere Ausgaben. Ein durchschnittlicher Artikel hat rund vier bis acht Schreibmaschinenseiten.

14. Hardcopys, Flußdiagramme, Zeichnungen und Bildschirmfotos dienen der Anschaulichkeit. Sie sollten nach Möglichkeit nicht fehlen. Zu jedem der vorgenannten »Zugaben« gehört aber eine Bildunterschrift und ein Verweis im Text.

15. Programme/Artikel die unserem Verlag zur Veröffentlichung angeboten werden, sollten aus urheberrechtlichen Gründen nicht gleichzeitig einem anderem Verlag vorliegen.

16. Das 64'er Magazin zahlt für Listings eine Pauschale zwischen 100 und 300 Mark. Für reine Artikel beträgt das Honorar zwischen 0,80 und 1,00 Mark pro Druckzeile. Für Disketten/Kassetten werden 30 Mark extra berechnet.

17. Sollten sich nach Erhalt eines positiven Anwortschreibens noch irgendwelche Änderungen oder Verbesserungen des Programms ergeben haben, teilen Sie uns das bitte umgehend mit. In diesem Falle benötigen wir ein vollständig neues Listing mit entsprechendem Datenträger.

(aa)

Die Ka-Di-Ecke

Mit unserer Kassetten-Disketten-Ecke bieten wir Ihnen im Rahmen des Leserservice die Alternative zum mühsamen Eintippen: Einfach Kassette bestellen und Programme laden.

Hier eine kleine Übersicht über den Lieferumfang unserer letzten Ausgaben (Preis jeweils 29,90 Mark pro Kassette). Zu den Programmen sind immer die Seitenzahlen angegeben, unter denen Sie die Beschreibungen im 64'er Magazin finden können. Abkürzungen: SB = Simons Basic wird benötigt; + 16K = es wird eine 16-KByte-Speichererweiterung benötigt; GV = Grundversion; +3 K = 3-KByte-Erweiterung; und so weiter. SE = Supererweiterung nur VC 20.

Die Programme auf Diskette zu liefern ist uns leider aus produktionstechnischen Gründen noch nicht möglich.

Commodore 64

Kassette zu Ausgabe 4/84:
- Drawline + Demos S. 65
- Sprite Move S. 70
- Invaders (SB) S. 74
- Cäsar S.78
- Disk-Copy S.92
- Merge S.94

Bestellnr. CB007

Kassette zu Ausgabe 5/84:
- Adress & Telefonregister S.64
- Fahrsimulator S.82
- Schatzsucher S.90

Bestellnr. CB008

Kassette zu Ausgabe 6/84:
- Lehrerkalender S.64
- Morsetrainer S.72
- Supervoc S.69
- Grafische Darstellg.(SB) S.82
- Hot Wheels S.98

Bestellnr. CB012

Kassette zu Ausgabe 7/84:
- Terminalprogramm S.24
- Programmverwaltung S.72
- Russvok (SB) S.76
- Crown No.1 S.80
- Space Invaders S.81
- 1520 Hardcopy S.108
- Centronics-Interface S.110
- Kurvendiskussion S.116
- Copy Rel. Datei S.132
- Autostart S.138
- Strubs (OP und QP) S.154

Bestellnr. CB009

VC 20

Kassette zu Ausgabe 4/84:
- Elektr.Notiz, + 16 K S.50
- Rennfahrer, GV S.86
- Erste Hilfe, GV o.> S.88
- Disk-Copy, +3 K o.> S.92

Bestellnr. VC006

Kassette zu Ausgabe 5/84:
- Relative Datei, +8 K S.69
- Schmatzer, GV S.76
- 3D-Grafik, +8 K S.78
- VC 20 Rallye, +28 K S.128

Bestellnr. VC007

Kassette zu Ausgabe 6/84:
- Movemaster, +8 K S.78
- Ghost Manor, GV S.104
- Logic DisAss, +3 K o.> S.108
- Underground, +16 K S.120

Bestellnr. VC008

Und nun noch eine kleine Bitte: Bevor Sie bei Nichtfunktionieren eines Programms bei uns anrufen, lesen Sie sich die Bedienungsanweisung in der entsprechenden Ausgabe vom 64'er durch.

PS: Bitte Bestellkarte am Ende des Heftes benutzen.



Falls Sie die bisherigen Ausgaben des 64'er-Magazins gelesen haben, sind Ihnen auch die bisher ausgeschriebenen Wettbewerbe bekannt. Wie ist der Stand? Wann werden die Sieger bekanntgegeben?

Der erste Wettbewerb — das schönste Sprite — ist bereits abgeschlossen (Ausgabe 7/84). Aber die restlichen Programmierwettbewerbe stehen noch offen. Da ist zum Beispiel das Kreuzworträtsel. Zugegeben, das ist wirklich eine harte Nuß. Ab und zu rufen Leser an und fragen nach dem Einsendeschluß. Da uns klar war, daß nicht jeder solch ein Programm in der Schublade liegen hat, verzichteten wir auf einen Termin. Aber jetzt sollten sich eventuelle Mitbewerber den letzten Stoß geben. In Heft 11 wird der erste Gewinner bekanntgegeben. Jeder, der jetzt noch, bis zum 15. September 1984, ein Kreuzworträtsel-Programm einschickt, hat allergrößte Chancen. Denn bisher ist erst 1 (ein) Programm bei uns eingetroffen! Aber das ist gar nicht so schlecht. Also, auf zum Endspurt.

Wo bleiben die Unterprogramme

Ein Wettbewerb ohne Einsendeschluß ist die Unterprogrammbibliothek. Denn dieser Wettbewerb soll eine ständige Einrichtung werden. Wir werden ab Ausgabe 10 jeden Monat ein eingeschicktes Unterprogramm prämieren. Zusätzlich veröffentlichen wir eine kleine Übersicht über bisher eingesandte Programme; damit nicht zum 30sten Male eine PRINT USING-Funktion eingeschickt wird. Und um vielleicht einige Anregungen zu geben. In diesem Zusammenhang noch eine Bitte: Leider versäumte ich den Hinweis, daß eine Kassette oder besser noch eine Diskette mit dem Unterprogramm und, wenn möglich mit einem Demoprogramm mitzuschicken ist. Es erleichtert uns die Arbeit und auch die Bewertung ganz enorm. Auch die Fixierung auf Basicprogramme war zu erkennen. Selbstverständlich sind auch Maschinenspracheroutinen erlaubt, ja, sogar erwünscht. Diese sollten allerdings von Basicprogrammen aufrufbar beziehungsweise ladbar sein. Auf dem mitgeschickten Datenträger (Kassette/Diskette) sollte das Maschinenprogramm, ein Basic-Lader (DATAs) und ein Demonstrationsprogramm gespeichert sein. Und die Länge eines Unterprogramms wird auch nicht festgelegt. Denn ein Unterprogramm muß nicht unbedingt sehr kurz sein.

Ich hoffe, daß damit einige Unklarheiten beseitigt wurden. Und nun viel Spaß bei der Entwicklung der Programme. (gk)

Wir suchen die Anwendung des Monats

Anwendung des Monats, was ist das? Nun, Sie haben einen Commodore 64 oder einen VC 20 und versuchen diesen irgendwie sinnvoll einzusetzen. Unter einer sinnvollen Anwendung versteht die 64'er Redaktion alles, was beispielsweise Programme im häuslichen Bereich bewirken. Es kann sich dabei um die Berechnung der Benzinkosten für Ihren Wagen handeln, um ein eigenes Textverarbeitungsprogramm gehen, sich um die Verwaltung Ihrer Tiefkühltruhe drehen oder ein ausgeklügeltes Telefon- und Adreßregister sein.

Setzen Sie Ihren VC 20/C 64 mehr oder weniger beruflich ein? Auch, oder vor allem, das ist eine sinnvolle Anwendung. Sie führen die Lohn- und Gehaltsabrechnung, Ihre Lagerverwaltung, die Bestellungen auf einem Commodore-Heimcomputer durch? So spezielle Anwendungen wie die Berechnung der Statik von selbstgezimmerten Regalen, von Klimadiagrammen oder Vokabellernprogrammen für den Schulunterricht oder die Zinsberechnung bei Krediten sind ebenfalls Themen, die mehr als konkurrenzfähig sind.

Uns ist die Anwendung des Monats

500 Mark

wert.

Schreiben Sie uns, was Sie mit Ihrem Computer machen:

Redaktion 64'er, Aktion: Anwendung des Monats, Hans-Pinsel-Str. 2, 8013 Haar bei München.

Programmierwettbewerb: Der beste Einzeiler

Sie haben richtig gelesen. Gesucht wird der beste Einzeiler. Auf jeden Fall darf es nur eine Basiczeile sein, in der Ihre Idee »Raum« fassen soll.

In der Kürze liegt die Würze, meinen wir. Probieren Sie das einmal aus. So schwer kann es ja nicht sein, eine einzige Zeile auf Papier zu bringen.

Aber bedenken Sie bitte: Die Konkurrenz wird groß sein. Jetzt werden sich sogar die Spitzenkönner, die sonst zu faul sind, ellenlange Programme zu schreiben, hinsetzen und nachdenken. Also gehen Sie nicht leichtfertig mit der einen Zeile um. Vielleicht fragen sich einige von Ihnen, was man denn wohl besonders Geistreiches in eine einzige Programmzeile setzen kann. Ein Beispiel dazu. Uns flatterten schon Programme auf den Tisch, die eine Dezimalzahl in eine Hexadezimalzahl umwandelten. Manche von Ihnen benötigten irre viel Programmspeicherplatz. Dieses Problem läßt sich aber auch in einer einzigen Zeile lösen! Stellen Sie sich vor, so ein Einzeiler wird Listing des Monats...

Seien Sie also kreativ. Scheuen Sie keine Tipparbeit, denn Sie werden eine Menge Papier verschwenden, bis Sie eine akzeptable Idee verwirklicht haben.

Schicken sie keine Kassette oder Diskette mit dem »Programm« ein. Schreiben Sie Ihre Lösung einfach auf ein Blatt Papier, dazu eine ausführliche Beschreibung. Sie wissen schon, mit Definition der verwendeten Variablen und so weiter. (Ein Flußdiagramm oder Struktogramm muß nicht unbedingt sein.) Und: Vergessen Sie die Adresse auf Ihrer Lösung nicht. Senden Sie Ihren Einzeiler dann an:

Markt & Technik Verlag
Redaktion 64'er,
Programmierwettbewerb:
Einzeiler
Hans-Pinsel-Str. 2,
8013 Haar bei München

Einmal im Monat gibt es die SUPERCHANCE

Diese nicht einmalige Gelegenheit sollten Sie nutzen. Wie? Schicken Sie uns Ihr bestes, selbst erstelltes Programm. Bei der Art des Programms sind wir nicht wählerisch.

Sie haben ein sehr gutes (Schieß-, Knobel-, Denk-, Action-, Abenteuer-) Spiel geschrieben: einschicken!

Sie verfügen über ein komfortables Disketten-Kopier-(Sortier-)-Programm mit einigen außergewöhnlichen Leistungsmerkmalen: einschicken!

Sie haben das Basic um einige sinnvolle Befehle erweitert: einschicken!

Sie arbeiten mit einem selbsterstellten Textverarbeitungsprogramm, einer eigenen Tabellenkalkulation, einem semiprofessionellen Datenverwaltungsprogramm: einschicken!

Sie zeichnen und konstruieren mit einem selbsterstellten Programm in hochauflösender Grafik: einschicken!

Wir freuen uns über jeden Beitrag und honorieren mit bis zu

2 000 Mark

für das Listing des Monats

Aus den besten Listings, die veröffentlicht werden, sucht die 64'er-Redaktion einmal im Monat das »Listing des Monats« aus. Alle Listings, die im 64'er abgedruckt sind, werden mit 100 bis 300 Mark honoriert. Die genaue Vorgehensweise beim Einsenden von Listings ist in »Wie schicke ich meine Programme ein?« Ausgabe 4/84 beschrieben:

Schicken Sie Ihr Listing an: Redaktion 64'er, Superchance: Listing des Monats, Hans-Pinsel-Str. 2, 8013 Haar bei München.

Programmierwettbewerb: Das lustigste Programm

Computer Programme für den VC 20 oder C 64 gibt es fast wie Sand am Meer. Die Palette reicht von der ernsthaften Anwendung wie Textverarbeitung oder Dateiverwaltung über Utilities, Hilfsprogramme zum Erstellen von Grafik und Sound, Programmierhilfen und Spracherweiterungen bis zu Spielen. Und auch Spiele kann man noch unterteilen in verschiedene Arten. So gibt es Actionspiele aber auch Denk- und Strategiespiele.

Relativ selten findet man jedoch echte Jux-Programme, solche, die einen zum Lachen bringen oder durch Gags verblüffen. Programme, die man zum Beispiel auf Partys vorführen kann. Aber das soll sich durch unseren Programmierwettbewerb ändern.

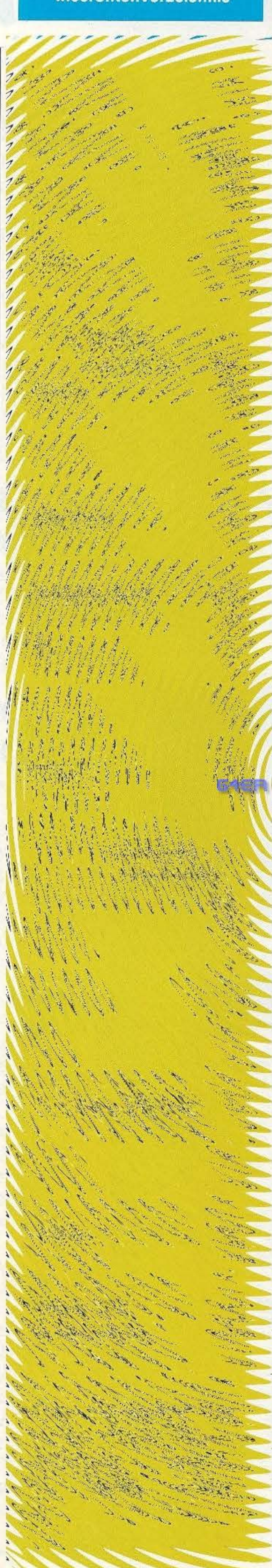
Insgesamt 1000 Mark zu gewinnen

Schreiben Sie ein solches Programm und schicken Sie es uns auf Diskette oder Kassette zu. Die Länge ist unwesentlich. Allein entscheidend ist die Wirkung.

Schicken Sie Ihre Programme auf Diskette/Kassette unter dem Stichwort »Programmierwettbewerb: Das lustigste Programm« an folgende Adresse:

Verlag Markt & Technik, Redaktion 64'er, Hans-Pinsel-Str. 2, 8013 Haar bei München.

Inserentenverzeichnis



Impressum

Herausgeber: Carl-Franz von Quadt, Otmar Weber

Chefredakteur: Michael M. Pauly (py)

Stellv. Chefredakteur: Michael Scharfenberger (sc)

Redakteure: aa = Albert Absmeier, leitender Redakteur (130), ev = Volker Everts (278), kg = Karin Gößlinghoff (269), gk = Georg Klinge (169), rg = Christian Rogge (278)

Redaktionsassistenz: Dagmar Zednik-Djadja (237)

Fotografie: Janos Feitser, Titelfoto: Alex Kempkens

Layout: Leo Eder (Ltg.), Dagmar Berninger, Willi Gründl, Walter Höß, Cornelia Weber

Auslandsrepräsentation:
Schweiz: Markt&Technik Vertriebs AG, Alpenstrasse 14, CH-6300 Zug, Tel. 042-22 31 55/56, Telex: 862 329 mut ch
USA: M&T Publishing, 2464 Embarcadero Way, Palo Alto, CA 94303; Tel. 001-4240600; Telex 752351

Manuskripteinsendungen: Manuskripte und Programmlistings werden gerne von der Redaktion angenommen. Sie müssen frei sein von Rechten Dritter. Sollten sie auch an anderer Stelle zur Veröffentlichung oder gewerblichen Nutzung angeboten werden, so muß dies angegeben werden. Mit der Einsendung von Manuskripten und Listings gibt der Verfasser die Zustimmung zum Abdruck in von der Markt & Technik Verlags AG herausgegebenen Publikationen und zur Vervielfältigung der Programmlistings auf Datenträger. Honorare nach Vereinbarung. Für unverlangt eingesandte Manuskripte und Listings wird keine Haftung übernommen.

Herstellung: Klaus Buck (180), Leo Eder (181)

Anzeigenleitung: Peter Schrödel (156)

Anzeigenverkauf: Alfred Reeb (211)

Anzeigenverwaltung und Disposition: Michaela Hörl (171)

Anzeigenformate: 1/1-Seite ist 266 Millimeter hoch und 185 Millimeter breit (3 Spalten à 58 mm oder 4 Spalten à 43 Millimeter). Vollformat 297 x 210 Millimeter. Beilagen und Beihefter siehe Anzeigenpreisliste.

Anzeigenpreise: Es gilt die Anzeigenpreisliste Nr. 1 vom 1. März 1984.

Anzeigengrundpreise: 1/1 Seite sw: DM 7400,-. Farbzuschlag: erste und zweite Zusatzfarbe aus Europaskala je DM 1000,-. Vierfarbzuschlag DM 3000,-. Plazierung innerhalb der redaktionellen Beiträge: Mindestgröße 1/4-Seite

Anzeigen im Einkaufs-Magazin: Die ermäßigten Preise im Einkaufs-Magazin gelten nur innerhalb des geschlossenen Anzeigenteils, der ohne redaktionelle Beiträge ist. 1/1-Seite sw: DM 5400,-. Farbzuschlag: erste und zweite Zusatzfarbe aus Europaskala je DM 1000,-. Vierfarbzuschlag DM 3000,-. **Anzeigen in der Fundgrube: Private Kleinanzeigen** mit maximal 5 Zeilen Text DM 5,- je Anzeige. **Gewerbliche Kleinanzeigen:** DM 10,- je Zeile Text.
Auf alle Anzeigenpreise wird die gesetzliche MwSt. jeweils zugerechnet.

Vertriebsleitung, Werbung: Hans Hörl (114)

Vertrieb Handelsauflage: Inland (Groß-, Einzel- und Bahnhofsbuchhandel) sowie Österreich und Schweiz: Pegasus Buch- und Zeitschriften-Vertriebs GmbH, Plieninger Straße 100, 7000 Stuttgart 80 (Möhringen), Telefon (07 11) 72 004-0

Erscheinungsweise: 64'er, Magazin für Computerfans, erscheint monatlich, Mitte des Vormonats.

Bezugsmöglichkeiten: Leser-Service: Telefon 089/46 13-1 19. Bestellungen nimmt der Verlag oder jede Buchhandlung entgegen. Das Abonnement verlängert sich zu den dann jeweils gültigen Bedingungen um ein Jahr, wenn es nicht zwei Monate vor Ablauf schriftlich gekündigt wird.

Bezugspreise: Das Einzelheft kostet DM 6,-. Der Abonnementspreis beträgt im Inland DM 72,- pro Jahr für 12 Ausgaben. Darin enthalten sind die gesetzliche Mehrwertsteuer und die Zustellgebühren. Der Abonnementspreis erhöht sich um DM 18,- für die Zustellung im Ausland, für die Luftpostzustellung in Ländergruppe 1 (z.B. USA) um DM 38,-, in Ländergruppe 2 (z.B. Hongkong) um DM 58,-, in Ländergruppe 3 (z.B. Australien) um DM 68,-.

Druck: Druckerei E. Schwend GmbH, Schmollerstr. 31, 7170 Schwäbisch Hall

Verantwortlich: Für redaktionellen Teil: Michael M. Pauly.
Für Anzeigen: Peter Schrödel.

Vorstand: Carl-Franz von Quadt, Otmar Weber

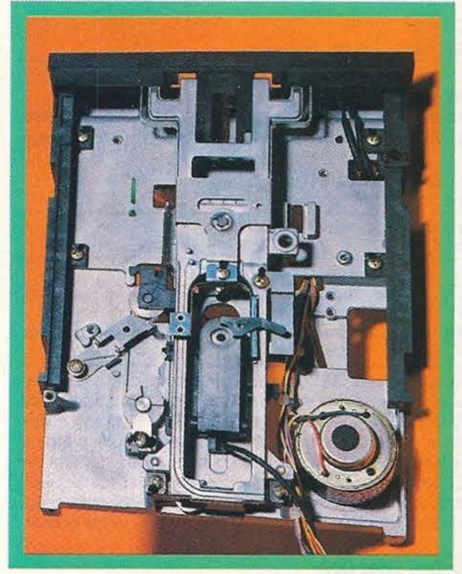
Anschrift für Verlag, Redaktion, Vertrieb, Anzeigenverwaltung und alle Verantwortlichen:
Markt&Technik Verlag Aktiengesellschaft, Hans-Pinsel-Straße 2, 8013 Haar bei München, Telefon 089/46 13-0, Telex 522 052

Telefon-Durchwahl im Verlag:

Wählen Sie direkt: Per Durchwahl erreichen Sie alle Abteilungen direkt. Sie wählen 0 89-46 13 und dann die Nummer, die in Klammern hinter dem jeweiligen Namen angegeben ist.







Lernen Sie die Floppy kennen

Die meisten VC 20- und C64-Anwender dürften mittlerweile statt einer Datasette ein Floppy-Laufwerk zu Hause stehen haben. In dieser ob ihrer Zuverlässigkeit und Geschwindigkeit oft geschmähten Diskettenstation steckt mehr, als man zunächst vermuten möchte. Mit dem neuen Floppy-Kurs steigen wir unter anderem in das Betriebssystem (eigener 6502-Prozessor mit 16 KByte ROM und 2 KByte RAM) ein. Sie erfahren alles über Ihre Floppy.

Drucker: Vergleichstest und Marktübersicht

Jeder Computeranwender steht früher oder später vor dem Problem: Welchen Drucker für mein Gerät (und für meinen Geldbeutel)? Unser Drucker-Vergleichstest soll allen Besitzern eines C 64 oder VC 20 eine Hilfestellung bei der Auswahl des richtigen Druckers sein. Unsere Marktübersicht informiert über die Vielfalt der an den Commodore anschließbaren Drucker.





Hardcopy-Routinen

Wir bringen endlich eine Sammlung von Hardcopy-Routinen für die gängigsten Drucker: MPS 801, VC 1515, MPS 802, VC 1526, Epson FX 80, Gemini 10X und als besonderen Leckerbissen eine farbige Hardcopy mit dem VC 1520-Plotter.

Vom Umgang mit Datex-P

Jeder an der Datenfernübertragung Interessierte spricht davon, doch keiner weiß genau Bescheid. Wir liefern eine praktische Anleitung zum Umgang mit Datex-P und ausländischen Netzwerken.

40/80-Zeichenkarte für VC 20

Der VC 20 kann tatsächlich wesentlich mehr, als man ihm im allgemeinen zutraut. Mit einer 40/80-Zeichenkarte ausgerüstet, taugt er sogar als Textverarbeitungssystem. Wir zeigen, was diese Erweiterung kann und wie man damit arbeitet.

Software-Test

— Vizawrite und Vizastar: Ein Textverarbeitungs- und ein Datenverwaltungsprogramm der gehobenen Kategorie
— ExDOS & Disk-Doctor: Wir haben uns ein sehr interessantes Disk-Utility-Programm für Sie angesehen
— Audiogenic-Forth: Eine Forth-Version für C 64/VC 20, die ohne Diskettenlaufwerk auskommt
— Loja: Mit diesem Programm wird die Lohn- und Einkommensteuererklärung zum Vergnügen.

Listing des Monats: Schneller laden

Das Disketten-Laufwerk VC 1541 ist sicherlich jedem zu langsam. Mit unserem Listing des Monats lassen sich fast alle Programme fünfmal schneller laden.

So laufen Basic-Programme schneller

Wir haben Ihnen in mehreren Ausgaben die strukturierte Programmierung nähergebracht. Dadurch werden die Basic-Programme aber auch furchtbar langsam. Mit einigen »Tricks« lassen sich diese Programme in der Abarbeitungsgeschwindigkeit um den Faktor zehn steigern.

Listings

— Anwendung des Monats: Finanzmathematik — ein Listing, das einige besondere Vorzüge bietet
— Userport-Tastatur: Taschenrechner als frei programmierbare Tastatur anschließen
— Diskettenorganisation: Alle Diskettenoperationen per Menü wählen
— Apocalypse now: Testen Sie Ihre Fähigkeiten als Hubschrauberpilot
— Epidemic: Ein strategisches Spiel für den VC 20
— Video-Vorspann: Ein kleines, aber nützliches Programm für alle Video-Amateure unter den VC 20-Besitzern
— und natürlich auch in der nächsten Ausgabe wieder viele Tips und Tricks für den VC 20 und den C 64




64ER ONLINE